APPENDIX K

SOURCE CODE FOR THE SFWMM V5.5

Version of SFWMM v5.5 has not been finalized pending review comments. However, the beta version, SFWMM v5.45, provided below.

abc.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: abc.inc,v 1.6 2003/07/03 20:07:53 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/abc.inc,v $
C       $Name: rel-5-5 $
C                                                                       -
C-----------------------------------------------------
      common /ABC/ STG_MIN_WS_DSCHG(max_ncnls),CHDEP(max_ncnls),
     &IBM,RCHG(max_ncells),COUTM(max_ncnls),QU(max_ncnls),MONTH,
     &OVLMAX(max_n_months),FRF,outflow_to_loxrv,POND(max_ncells),
     &DSWLM(max_ncnls),QUM(max_ncnls),NCA,IEM,ETM(max_ncells),
     &max_nca_ov,OVLFLO(max_ncells,2),H(max_ncells),
     &NPRTY(max_ncnls,max_nostr_cnl),BEGSTG(max_ncnls),capac_mcnl,
     &DRYSEAS,OFMC(max_n_lu_type,4),STRCF(max_ncnls,max_nostr_cnl),
     &IBEGYR,BSEFLW,BMPRED,IWBMAX,STAGEND(max_ncells),
     &SINFLTM(max_ncells),ISM,facts2mc,nca_alt,VINM(max_ncnls),IDAY,
     &ovlflo_to_canal(max_ncells),rmonthly_ovlflo_to_canal(max_ncells),
     &HURRPRED,idbsnopt1,LOXNOD1,LOXNOD2,ws_to_lec_s354,ws_to_lec_s351,
     &ws_to_lec_s352,nod_ds_s11ab,nod_ds_s11c,nod_ds_s9,no_flwth_cnls,
     &nod_ds_s150,diff_runoff_bmp(max_n_eaa_condt,366),
     &flow_reduction_bmp(max_n_eaa_condt),NINOSEAS,
     &excess_reduction(max_n_eaa_condt),
     &stg_trigger_fl(max_ncnls,max_nostr_cnl,5),itylsl,fallseas,
     &stg_min_fc(max_ncnls,max_nostr_cnl,5),
     &n_trigger_fc(max_ncnls,max_nostr_cnl),il31w_min_rf,iuse_lok1,
     &n_stg_min_cnl(max_ncnls,max_nostr_cnl),NODE_G3273,NCNL_WS_MIN,
     &IFLS174,iuse_lok2,IFLS338,icnl_flwth(max_n_flwth_cnls),
     &NDS_CNL_NO(30),CGINFM(max_ncnls),
     &icurtw_opt(max_ncells,max_n_seas,4),ICURTAIN(max_ncells),NCA_OV,
     &DTS,NTSTEP_OV,flow_to_l8sta,NCL_NO(30),
     &frac_runoff_asr(max_n_eaa_condt),iasr_opt_eaa(max_n_eaa_condt),
     &itribres2,tot_avl_l8_bpit_res,frac_avail_asr_to_meet_dmnd(5),
     &ieaa_res_asr(max_n_eaa_condt),
     &idbsnopt2,frac_rem_capac(max_n_eaa_condt),ws_to_lec_s351_nnrc,
     &ws_to_lec_s351_hill,ws_to_lec_s351_wca3a,
     &ires_opt_eaa(max_n_eaa_condt),ieaa_res_asr_reg(max_n_eaa_condt,5),
     &seep_to_canal(max_ncells),rmonthly_seep_to_canal(max_ncells),
```

```
     &DQU_ORIG(max_ncnls),canal_evap(max_ncells),
     &rmonthly_canal_evap(max_ncells),iserv_area_loc(max_ncnls),
     &icnl_fc_flwth(max_n_flwth_cnls),no_fc_flwth_cnls,
     &frac_reg_flow_nesrs_s355,adjust_for_area_n(max_ncells),
     &adjust_for_area_m1(max_ncells),deten_c(max_ncells),
     &deten_defc(max_n_lu_type),frac_bkflw_wpb_via_l8_abv_flr,
     &yval(366),yval1(366),yval2(366),yval3(366),yval4(366),yval5(366),
     &yval6(366)
C-----------------
        LOGICAL DRYSEAS,HURRPRED,NINOSEAS,fallseas
        DOUBLE PRECISION chdep,stg_min_ws_dschg
```

accum_estuar_dmnds.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: accum_estuar_dmnds.F,v 1.8 2005/06/10 13:28:58 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/accum_estuar_dmnds.F,v $
C   $Name: rel-5-5 $
C
C-----------------------------------------------------------------------------
C   SUBROUTINE ACCUMULATES ESTUARINE DEMANDS AND CALCULATES SUPPLEMENTAL
C   RELEASES FROM LAKE OKEECHOBEE TO MEET THE DEMANDS FOR CALOOSAHATCHEE
C   AND ST LUCIE ESTUARIES FOR EVERY MONTH OF CURRENT YEAR IN
C   SIMULATION.  MAXIMUM AVAILABLE VOLUMES OF WATER FOR BACKFLOW INTO
C   LAKE OKEECHOBEE FROM CALOOSAHATCHEE/ST LUCIE BASINS ARE ALSO CALCULATED
C   HERE FOR EVERY MONRH OF CURRENT YEAR.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE accum_estuar_dmnds (numd, bflo_cap0_c43est,
     & lok_supp_c43est, lok_supp_day_c43est, lok_supp_c44est,
     & bflo_cap0_c44est, bsn2est1, bsn2est2, cale_dmd_day, sle_dmd_day,
     & stltrro, dmndro, ifyr, iyear)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'

      DIMENSION numd(12)

      REAL bflo_cap0_c43est(12), bflo_cap0_c44est(12), bsn2est1(12,31),
     & bsn2est2(12,31), cal_basin_ro(12), cale_dmd_day(366),
     & cale_dmd_mo(12), cb_ro(31), dmndro(8,2,2,366),
     & lok_supp_c43est(12), lok_supp_day_c43est, lok_supp_c44est(12),
     & other_ro(12), sl_basin_ro(12), sl_ro(31), sle_dmd_day(366),
     & sle_dmd_mo(12), stltrro(366)
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C accumulate estuarine demands and calculate LOK supplemental release
C and maximum backflow capacity for every month of this year

      IF (demand_level_opt .eq. 'CURR') THEN
         k = 1
      ELSE
         k = 2
      ENDIF

      idayno = 0
      numd(2) = 28

      IF (mod(ifyr+iyear-1,4) .eq. 0) numd(2) = 29

      DO imo = 1,12

         sle_dmd_mo(imo) = 0.0
         sl_basin_ro(imo) = 0.0
         cale_dmd_mo(imo) = 0.0
         cal_basin_ro(imo) = 0.0
         other_ro(imo) = 0.0

         DO ida = 1,numd(imo)
            idayno = idayno + 1
            IF (simsles .ne. 'NO ') THEN
c for st lucie
               sle_dmd_mo(imo) = sle_dmd_mo(imo) + sle_dmd_day(idayno)
               sl_basin_ro(imo) = sl_basin_ro(imo) + dmndro(1,1,k,
     $         idayno) + stltrro(idayno)
c
               sl_ro(ida) = dmndro(1,1,k,idayno)
               other_ro(imo) = other_ro(imo) + stltrro(idayno)
c
            ENDIF
            IF (simcaes .ne. 'NO ') THEN
c for Caloosahatchee
               cale_dmd_mo(imo) = cale_dmd_mo(imo) + cale_dmd_day
     $         (idayno)
               cal_basin_ro(imo) = cal_basin_ro(imo) + dmndro(2,1,k,
     $         idayno)
c
               cb_ro(ida) = dmndro(2,1,k,idayno)
c
            ENDIF
         END DO             !  end of day loop
c
         IF (simsles .ne. 'NO ') THEN
c for St. Lucie
            IF (sle_dmd_mo(imo) .gt. sl_basin_ro(imo)) THEN
               lok_supp_c44est(imo) = sle_dmd_mo(imo) - sl_basin_ro(imo)

               factor = 1.0
            ELSE
               lok_supp_c44est(imo) = 0.0
               IF (sl_basin_ro(imo) .le. 0.) THEN
```

```
                  factor = 1.0
               ELSE IF(sle_dmd_mo(imo) .gt. other_ro(imo)) THEN
                  factor = (sle_dmd_mo(imo) - other_ro(imo)) /
     $            (sl_basin_ro(imo) - other_ro(imo))
                  factor = amin1(1.0,factor)
               ELSE
c non-c44 runoff is more than enough to meet estuarine demand
                  factor = 0.0
               ENDIF
            ENDIF
c
            DO ida = 1,numd(imo)
               bsn2est2(imo,ida) = factor*sl_ro(ida)
            END DO
c
            IF (sle_dmd_mo(imo) .le. 0.001) THEN
               bflo_cap0_c44est(imo) = 999999.0
            ELSE
               IF (sl_basin_ro(imo) .gt. sle_dmd_mo(imo)) THEN
                  bflo_cap0_c44est(imo) = sl_basin_ro(imo) - sle_dmd_mo
     $            (imo)
               ELSE
                  bflo_cap0_c44est(imo) = 0.0
               ENDIF
            ENDIF
         ENDIF
c
         IF (simcaes .ne. 'NO ') THEN
c for Caloosahatchee
            IF (cale_dmd_mo(imo).gt.cal_basin_ro(imo)) THEN
               lok_supp_c43est(imo) = cale_dmd_mo(imo) - cal_basin_ro
     $         (imo)
               factor = 1.0
c
            ELSE
               lok_supp_c43est(imo) = 0.0
               IF (cal_basin_ro(imo).le.0.) THEN
                  factor = 1.0
               ELSE
                  factor = cale_dmd_mo(imo)/cal_basin_ro(imo)
               ENDIF
            ENDIF
c
            DO ida = 1,numd(imo)
               bsn2est1(imo,ida) = factor*cb_ro(ida)
            END DO
c
            IF (cale_dmd_mo(imo).le.0.001) THEN
               bflo_cap0_c43est(imo) = 999999.0
            ELSE
               IF (cal_basin_ro(imo).gt.cale_dmd_mo(imo)) THEN
                  bflo_cap0_c43est(imo) = cal_basin_ro(imo) -
     $            cale_dmd_mo(imo)
               ELSE
                  bflo_cap0_c43est(imo) = 0.0
               ENDIF
            ENDIF
```

```
         ENDIF
c
      END DO                 !  end of month loop
c
      RETURN
      END
```

add_misc.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: add_misc.inc,v 1.4 2003/07/03 20:07:53 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/add_misc.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      common /add_misc/ n_add_str_dep(max_n_str_lok_in_wca),
     &type_flow_s(max_n_str_lok_in_wca),
     &node_add(max_n_str_lok_in_wca,max_add_grid_loc),
     &n_add_can_dep(max_n_str_lok_in_wca),
     &n_add_grid_loc(max_n_str_lok_in_wca),hinit(max_ncells),
     &pond0(max_ncells),stagelo0,chdep0(max_ncnls),
     &initialize_annually_opt,ifreq_init,solmx0(max_ncells)
C-----------------
      CHARACTER type_flow_s*4
```

addlok.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: addlok.inc,v 1.6 2003/07/03 20:07:54 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/addlok.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      common /addlok/ values(max_ncells),values_w_lok(max_ncells_w_lok),
     &isumlok(max_n_rows_lok+1)
```

agarea.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: agarea.F,v 1.21 2005/06/10 13:28:58 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/agarea.F,v $
C   $Name: rel-5-5 $
C
C
C  THIS ROUTINE SIMULATES THE WATER TABLE MANAGEMENT PRACTICES IN THE AG AREA BY
C  FORCING THE SOIL MOISTURE TO RESIDE WITHIN A VERY NARROW RANGE. EXCESS WATER
IS
C  REMOVED FROM THE SOIL AND ROUTED THOUGH THE OUTLET STRUCTURES OR TO A
RESERVOIR.
C  SUPPLEMENTAL IRRIGATION REQUIREMENTS ARE MET FROM APPROPRIATE SOURCES (LAKE
C  OKEECHOBEE OR RESERVOIR OR BOTH) AND ADDED TO THE SOIL COLUMN.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE agarea (jday, solmdph, kflo0, dmds4, dmds236, rain,
     & agrunoff_to_l8, dmnd_met_by_rain, dmnd_met_by_local_storage,
     & total_eaa_dmnd, total_et_red, agrunoff_to_l8w, agrunoff_to_l8s,
     & isem_flg, dmnd_sugh_orig, supp_eaa_dmnd, sum_eaa_dmnd_ssm,
     & alocnnr_potdel, pot_del_to_eaa_basin, sum_alocnnr, indx_lok_del,
     & total_small_afact, dmndro, itypern,struc_name_meas,
     & total_area_298d_grid)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'lakwca.inc'
```

```
      INCLUDE 'simflows.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'STRCOUNT.inc'

      REAL alocnnr_potdel(2), k, kc, et_red(max_ncells),
     & kflo0(max_n_struc_flw,366),
     & pot_del_to_eaa_basin(max_n_eaa_condt,2),
     & rain(max_ncells), rem_gwdpth(max_ncells), solmdph(max_ncells),
     & solprev(max_ncells), sum_eaa_dmnd_ssm(max_n_eaa_condt),
     & tdmnd(max_n_eaa_condt), total_small_afact(max_ncells),
     & trunoff(max_n_eaa_condt), vol_of_deliv(3) ,
     & excess_water_to_cell(max_ncells)
      CHARACTER struc_name_meas(max_n_struc_meas)*6

      DIMENSION  dmndro(8,2,2,366)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      DO node = 1,max_ncells
         excess_water_to_cell(node) = 0.0
      ENDDO
c
c capture boundary flow values for current time step
c
      DO i=1,nflpts
         is = istr_index(i)
         kflo(is) = kflo0(is,jday)
      ENDDO

c
c  route appropriate boundary flows into the grid cells of the
c  recipient canal basin in EAA
c
      DO 30 I = 1,NFLPTS
         IS = ISTR_INDEX(I)
         IF (FLNM(IS) .EQ. 'G136  ') THEN
c  route G136 flows from Western Basins
            if (istaopt(8) .eq. 1) then
               rinflow_g136 = G136EAA
            else
               rinflow_g136 = KFLO(IS)
            endif
            DO 10 IY = MAXY,42,-1
               DO 10 J = ISUM(IY) + 1,ISUM(IY + 1)
                  IF (IBSN(J) .EQ. 7.AND.land_use_type(LUTYP(J)).NE.
     $            'WETLAND') then
                     POND(J) = POND(J) + rinflow_g136 *86400./(NAGNODE
     $               (IBSN(J)-6)*GDAR*(1.0-carfact(j)
     $               -total_small_afact(j)))
                  ENDIF
   10       CONTINUE
         ELSEIF (FLNM(IS) .EQ. 'G88   ' .and. istaopt(6) .eq. 0) THEN
c  route G88 flows if STA-6 does not exist in simulation
            DO 20 IY = MAXY,42,-1
```

```
               DO 20 J = ISUM(IY) + 1,ISUM(IY + 1)
                  IF (IBSN(J) .EQ. 7.AND.land_use_type(LUTYP(J)).NE.
     $            'WETLAND') then
                     POND(J) = POND(J) + KFLO(IS) *86400./(NAGNODE
     $               (IBSN(J)-6)*GDAR*(1.0-carfact(j)
     $               -total_small_afact(j)))
                  ENDIF
   20       CONTINUE
         ENDIF
   30 CONTINUE
C
      DO 40 I = 1,NEAABSN
         AGP(I) = 0.0
         sum_avail_stor(i) = 0.0
         trunoff(i) = 0.0
         tdmnd(i) = 0.0
   40 CONTINUE
C
      dmnd_met_by_rain = 0.0
      dmnd_met_by_local_storage = 0.0
      total_dmnd_for_298d_grid = 0.0
      total_eaa_dmnd = 0.0
      DMDSUGH = 0.0
      ALOCNNR = 0.0
      DMNDAGL8 = 0.0
c      TAREAS4 = 36559.
c New irrigated LU for S236 (all in acres); see LU 03/25/03 kkonyha
c    citrus - 27 , cane - 7983 , veg - 414 , irr. pasture - 1886
c    total = 10,310 (old value pre-v5.0 10,346)
      TAREAS236 = 10310.
      sum_avail_l8ag = 0.0
      sum_avail_sugh = 0.0
      sum_avail_stor_eaa = 0.0
C
      DO 60 I = MAXY,42,-1
         DO 50 J = ISUM(I) + 1,ISUM(I+1)
            IF (runmode .eq. 'SIMUL' .OR. IBSN(J) .EQ.  10.OR.IBSN(J)
     $      .EQ. 40) THEN
               IF ((IBSN(J).LT.7.OR.IBSN(J).GT.10) .AND. IBSN(J) .NE.
     $          40.AND. IBSN(J) .NE. 42) GO TO 50
               IAR = IBSN(J) - 6
               ITYP = LUTYP(J)
C
C  Compute ET Coefficients for each Land use type in EAA
C
               K = ETK(ITYP,JDAY)
               IF (LUTYP(J) .EQ. 7) THEN
                  KC = K * ADJCFF(MONTH)*et_cal_adj_fact
     $            (lutyp(j),MONTH)
               ELSEIF(LUTYP(J) .EQ. 8) THEN
                  KC = K * ADJCFF(MONTH)*et_cal_adj_fact(lutyp(j)
     $            ,MONTH)
               ELSEIF(LUTYP(J) .EQ. 9) THEN
                  KC = K * ADJCFF(MONTH)*et_cal_adj_fact(lutyp(j)
     $            ,MONTH)
               ELSE
                  KC = K*et_cal_adj_fact(lutyp(j),MONTH)
```

```
               ENDIF
C
               ET = KC * potet(ietzon(j))/12.
               ETMX = ET
               GWD = ELLS(J) - H(J)
C
C  ET From Saturated Zone (Water Table)
C
               IF (GWD .GE. DRZ(LUTYP(J))) THEN
                  ET = 0.0
               ELSEIF (GWD .GT.SRZ(LUTYP(J))) THEN
                  ET = ET * (DRZ(LUTYP(J)) - GWD)/(DRZ(LUTYP(J))-SRZ
     $            (LUTYP(J)))
               ENDIF
C
C  ET From Unsaturated Zone (from soil moisture column) which
C  corresponds to crop requirements.
c
               IF (land_use_type(ITYP) .NE. 'WETLAND') THEN
                  solprev(j) = solmx(j)
                  IF (ETMX .LT. POND(J)+SOLMX(J)) THEN
                     ETU = ETMX
                  ELSE
                     ETU = POND(J) + SOLMX(J)
                  ENDIF
                  RCHG(J) = RCHG(J) - AMIN1(ET,ETMX-ETU)
C
                  et_cnl=kmax(lutyp(j))*potet(ietzon(j))/12.0
                  if (use_lec_et.ne.0) then
                     et_unsat_actual(J) = ETU
                     et_sat(J) = AMIN1(ET,ETMX-ETU)
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     et_ponding(J) = et_cnl * carfact(j)
                  endif
C
                  ETM(J) = ETM(J) + (ETU + AMIN1(ET,ETMX-ETU))
     $            * (1.0-total_small_afact(j)-carfact(j))
     $            + et_cnl * carfact(j)
c
c get the monthly reduction in et, if any, for each cell
c et reduction = max et - actual et
                  et_red(j) = etmx - (etu + amin1(et,etmx-etu))
c sum it up later for those EAA cells that are AG
C
c  determine the total demand in the eaa which is the
c  maximum ET (ETMX = k * PET)
c
C  total_eaa_dmnd includes Miami, NNR-Hill, WPB basins + Sugar Ranch

                 if ( ibsn(j) .ne. 40 .and. ibsn(j) .ne. 10 ) then
                     total_eaa_dmnd = total_eaa_dmnd + etmx * gdar
     $               *(1.0 - total_small_afact(j) - carfact(j))
     $               *1.9835/86400.
                 endif
c
                  USATDPH = SOLMDPH(J) - SOLMX(J)
                  IF (POND(J) - ETU .GT. USATDPH) THEN
```

```
                     POND(J) = POND(J) - ETU - USATDPH
                     SOLMX(J) = SOLMDPH(J)
                  ELSE
                     SOLMX(J) = SOLMX(J) + POND(J) - ETU
                     POND(J) = AMAX1(SOLMX(J)-SOLMDPH(J),0.0)
                     SOLMX(J) = SOLMX(J) - POND(J)
                  ENDIF
c runoff from irrigated cells
                  IF (SOLMX(J)+POND(J).GT.SOLCRNF(J,JDAY)) THEN
                     DPTHRNFF = SOLMX(J) + POND(J) - SOLCRNF(J,JDAY)
                  ELSE
                     DPTHRNFF = 0.0
                  ENDIF
               ELSE
c  runoff from wetland cells
                  GWDPTH = (ELLS(J) - H(J))*S(J)
                  IF (POND(J) .GE. GWDPTH) THEN
                     stg_node = POND(J) - GWDPTH + ELLS(J)
                  ELSE
                     stg_node = (POND(J)-GWDPTH)/S(J) + ELLS(J)

                  ENDIF
                  IF (amax1(stg_node-ELLS(J),0.0) - ET .GT. DETEN(J))
     $            THEN
                     DPTHRNFF = AMIN1(stg_node-ELLS(J)-ET-DETEN(J),
     $               0.0207)
                  ELSE
                     DPTHRNFF = 0.0
                  ENDIF
               ENDIF
               IF (land_use_type(lutyp(j)) .ne. 'WETLAND') THEN
c
c   portion of excess water used to recharge water table first if water table
c   is below the base of the unsaturated zone or soil column
c
                  GWMAXDP = ELLS(J) - SOLMDPH(J)/S(J) - H(J)
                  GWMAXDP = AMAX1(GWMAXDP,0.0) * S(J) + AMIN1(ET,ETMX
     $            -ETU)
                  ADDPTH = AMIN1(GWMAXDP,DPTHRNFF)
                  rem_gwdpth(j) = gwmaxdp - addpth
                  RCHG(J) = RCHG(J) + ADDPTH
                  IF (ADDPTH .GE. POND(J)) THEN
                     SOLMX(J) = SOLMX(J) - (ADDPTH-POND(J))
                     POND(J) = 0.0
                  ELSE
                     POND(J) = POND(J) - ADDPTH
                  ENDIF
                  DPTHRNFF = DPTHRNFF - ADDPTH
               ENDIF
               IF (DPTHRNFF .GT. 0.0) THEN
                  dph = 0.0
                  IF (land_use_type(LUTYP(J)) .NE. 'WETLAND') THEN
                     SOLMX(J) = SOLMX(J) + POND(J) - DPTHRNFF
                     POND(J) = AMAX1(SOLMX(J)-SOLMDPH(J),0.0)
                     SOLMX(J) = SOLMX(J) - POND(J)
c
c  whenever there is runoff, demand is solely met by rainfall
```

```
c  since rainfall is source of runoff
c
                     if (ibsn(j) .ne.  40 .and. ibsn(j) .ne. 10 ) then
                        dmnd_met_by_rain = dmnd_met_by_rain + etmx *
     $                  gdar * (1.0 - total_small_afact(j)
     $                  - carfact(j))/43560.
                     endif
c
                  ELSE
                     POND(J) = POND(J) - DPTHRNFF
                  ENDIF
c
                  IF (IBSN(J) .NE.  40.AND. IBSN(J) .NE. 42) THEN
c
c  sum volume of excess water within the majojor EAA canal basins
c  and 298 Districts
c
                     AGP(IAR) = AGP(IAR) - DPTHRNFF * GDAR
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     trunoff(iar) = trunoff(iar) + dpthrnff * gdar
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     if(ibsn(j) .eq. 10) then
                        BAGP(IAR) = BAGP(IAR) - DPTHRNFF * GDAR
     $                  * (1.0 - total_small_afact(j) - carfact(j))
                     endif
                     if (IBSN(J) .EQ. 10) THEN
                        total_dmnd_for_298d_grid
     $                  =total_dmnd_for_298d_grid - DPTHRNFF * GDAR
     $                  * (1.0 - total_small_afact(j) - carfact(j))
                     endif
                  ELSEIF (IBSN(J) .EQ. 40) THEN
c  sum volume of excess water within the ag areas in L-8 basin
                     DMNDAGL8 = DMNDAGL8 - DPTHRNFF * GDAR
     $               * (1.0 - total_small_afact(j) - carfact(j))
                  ELSEIF (IBSN(J) .EQ. 42) THEN
c  sum volume of excess water within the US Sugar Ranch
                     DMDSUGH = DMDSUGH - DPTHRNFF * GDAR
     $               * (1.0 - total_small_afact(j) - carfact(j))
                  ENDIF
C
                  IF ( I .GE. MINYNNR .AND. I .LE. MAXYNNR) THEN
                     IB = I - MINYNNR + 1
                     ICOL_NNR = J - ISUM(I) + MINX(I) - 1
                     IF (ICOL_NNR .GT. IXMNNNR(IB) .AND. ICOL_NNR .LE.
     $               IXMXNNR(IB)) THEN
                        ALOCNNR = ALOCNNR - DPTHRNFF * GDAR
     $                  * (1.0 - total_small_afact(j) - carfact(j))
                     ENDIF
                  ENDIF
               ELSE
                  dpthrnff = 0.0
                  IF (land_use_type(lutyp(j)) .ne. 'WETLAND') THEN
C
                     IF (land_use_type(lutyp(j)) .eq. 'AGRICUL') then
c compute the total et reduction for the eaa basin AG cells,
c excluding basin 40.
                        if (ibsn(j) .ne. 40 .and. ibsn(j) .ne. 10 ) then
```

```
                         total_et_red = total_et_red + et_red(j) *
     $                   gdar*(1.0-total_small_afact(j)-carfact(j))
     $                   /43560.
                      endif
                   ENDIF
c
                   if (ibsn(j) .eq. 40) rem_gwdpth(j) = 0.0
c
                   depth_below_min = amax1(solcrt(j,jday) - solmx(j),
     $             0.0)
                   DPH = depth_below_min + rem_gwdpth(j)
                   DPH = AMIN1(DPH,amax1(etmx-rain(j)/12.,0.0))
c
c  determine portion of demand in eaa met by rainfall and/or met
c  by local storage when no runoff is occurring
c
                   if (ibsn(j) .ne. 40 .and. ibsn(j) .ne. 10) then
                      if (dph .eq. 0.0) then
                         if (solmx(j) .ge. solprev(j)) then
                            dmnd_met_by_rain = dmnd_met_by_rain + etmx
     $                      *gdar* (1.0-total_small_afact(j)
     $                      -carfact(j))/43560.
                         else
c                            dmnd_met_by_local_storage =
c     $                      dmnd_met_by_local_storage + (etmx - rain
c     $                      (j)/12.)*gdar*(1.0 - total_small_afact(j)
c     $                      -carfact(j))/43560.
                            dmnd_met_by_rain = dmnd_met_by_rain + rain
     $                      (j)/12. * gdar* (1.0-total_small_afact(j)
     $                      -carfact(j))/43560.
                         endif
                      else
                         dmnd_met_by_rain = dmnd_met_by_rain + amin1
     $                   (rain(j)/12. ,etmx)*gdar* (1.0 -
     $                   total_small_afact(j)-carfact(j))/ 43560.
c                         solmx_pres = amax1(solmx(j),solcrt(j,jday))
c                         dmnd_met_by_local_storage =
c     $                   dmnd_met_by_local_storage + amax1(solprev(j)
c     $                   - solmx_pres,0.0) * gdar* (1.0 -
c     $                   total_small_afact(j)-carfact(j))/43560.
                      endif
                   endif
c
                   IF (IBSN(J) .NE.  40.AND. IBSN(J) .NE. 42) THEN
c  sum volume of supplemental demand for major EAA canal basins and
c  298 Districts
                      AGP(IAR) = AGP(IAR) + DPH * GDAR *
     $                (1.0 - total_small_afact(j) - carfact(j))
c                     IF (IBSN(J).EQ. 7) print *,J,DPH,AGP(IAR)/86400.
c    $                ,'  NNN ',solmx(j),depth_below_min,rem_gwdpth(j)
                      if (IBSN(J).EQ. 10) THEN
                         total_dmnd_for_298d_grid
     $                   =total_dmnd_for_298d_grid + DPH * GDAR
     $                   * (1.0-total_small_afact(j)-carfact(j))
                      endif
                      tdmnd(iar) = tdmnd(iar) + dph * gdar
     $                * (1.0-total_small_afact(j)-carfact(j))
```

```
                        BAGP(IAR) = BAGP(IAR) + DPH * GDAR
     $                  * (1.0-total_small_afact(j)-carfact(j))
                     ELSE
C
C   DEMAND FOR AGRICULTURAL AREA IN L-8 BASIN AND SUGAR PLANT
C
                        IF (IBSN(J) .EQ. 40) THEN
                           DMNDAGL8 = DMNDAGL8 + DPH * GDAR
     $                     * (1.0-total_small_afact(j)-carfact(j))
                        ELSE
                           DMDSUGH = DMDSUGH + DPH * GDAR
     $                     * (1.0-total_small_afact(j)-carfact(j))
                           tdmnd(is8bsn-6) = tdmnd(is8bsn-6) + dph
     $                     * (1.0-total_small_afact(j)-carfact(j))*gdar
                        ENDIF
                     ENDIF
C
                     IF ( I .GE. MINYNNR .AND. I .LE. MAXYNNR) THEN
                        IB = I - MINYNNR + 1
                        ICOL_NNR = J - ISUM(I) + MINX(I) - 1
                        IF (ICOL_NNR .GT. IXMNNNR(IB) .AND. ICOL_NNR
     $                  .LE. IXMXNNR(IB)) THEN
                           ALOCNNR = ALOCNNR + DPH * GDAR
     $                     * (1.0-total_small_afact(j)-carfact(j))
                        ENDIF
                     ENDIF
                     IF (IBSN(J) .NE. 7 .AND. IBSN(J) .NE. 8 .AND.
     $               IBSN(J) .NE. 9) THEN
                        SOLMX(J) = SOLMX(J) + amin1(depth_below_min,dph)
                        perc_to_wt = amax1(dph - depth_below_min,0.0)
                        RCHG(J) = RCHG(J) + perc_to_wt
                        rem_gwdpth(j) = rem_gwdpth(j) - perc_to_wt
                     ENDIF
                  ENDIF
               ENDIF
c
c   ET from small reservoir(s) in EAA
c
               DO ir = 1,noresincell(j)
                  iresnum = ires_index_in_cell(j,ir)
                  IF (sfactor(iresnum).LT.sfactmin.and.ires_small_sim
     $            (iresnum) .eq. 'YES') THEN
                    IF (land_use_type(LUTYP(J)) .ne. 'WETLAND') then
                     depth_of_deficit = 0.0
                     call etcomp(-1,ET,J,LUTYP_IN_RES(J,ir),JDAY ,
     $               depth_of_deficit,stage_res ,ET_outside_res,
     $               iresnum,ir,sfactor(iresnum),ells_in_res,1)
                     IF (pond_in_res(j,ir) .GT. ET) then
                        pond_in_res(j,ir) = pond_in_res(j,ir)
     $                  - ET
                        stage_res(j,ir) = ells_in_res(j,ir) +
     $                  pond_in_res(j,ir)
                     ELSE
                        stage_res(j,ir) = stage_res(j,ir)
     $                  - (ET - pond_in_res(j,ir))/s(j)
     $                  - pond_in_res(j,ir)
                        pond_in_res(j,ir) = 0.0
```

```
                    ENDIF
                   ENDIF
                 ENDIF
              ENDDO
           ENDIF
C-----------------
   50    CONTINUE
   60 CONTINUE
c
c  adjust total net dmnd in miami canal basin for sugar plantation
c
c      if (istaopt(10) .ne. 1) then
         if (nint(dmdsugh) .gt. 0) then
            agp(is8bsn-6) = agp(is8bsn-6) + dmdsugh
         endif
c      endif
c
c  take into account afficiency factor in removing excess water
c  from EAA - particularly in lowering water table
c
      if ( runmode .eq. 'SIMUL') then
         do ib = 1,neaabsn-1
            if (agp(ib) .lt. 0.0) then
               do i = 1,nthresholds(ib)
                  if(abs(agp(ib))/86400. .lt. runoff_thres(ib,i)) then
                     pct_daily_removed = pct_daily_runoff(ib,i-1) +
     $               ((abs(agp (ib)) /86400.-runoff_thres(ib,i-1))
     $               /(runoff_thres(ib,i) - runoff_thres(ib,i-1))) *
     $               (pct_daily_runoff(ib,i) - pct_daily_runoff(ib,i-1))
                     stor_eaa_adj(ib)=abs(agp(ib))*(1.0
     $               -pct_daily_removed) / 86400.
                     agp(ib) = agp(ib) * pct_daily_removed
                     bagp(ib) = bagp(ib) + agp(ib)
                     go to 70
                  endif
               enddo
            else
               stor_eaa_adj(ib)=0.0
            endif
   70       continue
         enddo
      endif
C
C    ROUTE RUNOFF FROM AGRIC AREA IN L-8 BASIN TO L-8
C
      agrunoff_to_l8 = 0.0
      IF (DMNDAGL8 .LT. -0.00001) THEN
         DMNDAGCFS = DMNDAGL8/86400.
         IF (.NOT. opt_l8_prop) THEN
            QU(IL8) = QU(IL8) + AMIN1(ABS(DMNDAGCFS),RL8AGRFCAP) *
     $      86400.
            agrunoff_to_l8 = amin1(ABS(DMNDAGCFS),RL8AGRFCAP)
            IF (ABS(DMNDAGCFS) .GT. RL8AGRFCAP) THEN
               IAGFLO(IAG(IL8AGBSN),1) =NINT(ABS(DMNDAGCFS)-RL8AGRFCAP)
            ENDIF
         ELSE
            QU(IL8W) = QU(IL8W) + AMIN1(ABS(DMNDAGCFS)/2.0,RL8AGRFCAP) *
```

```
     $      86400.
            agrunoff_to_l8w = AMIN1(ABS(DMNDAGCFS)/2.0,RL8AGRFCAP)
            RUNFF_EXCESS = AMAX1(ABS(DMNDAGCFS)/2. 0- RL8AGRFCAP,0.0)
            RL8AGCP = RL8AGRFCAP - AMIN1(ABS(DMNDAGCFS)/2.0,RL8AGRFCAP)
            QU(IL8S) = QU(IL8S) + AMIN1(ABS(DMNDAGCFS)/2.0,RL8AGCP) *
     $      86400.
            agrunoff_to_l8s = amin1(ABS(DMNDAGCFS)/2.0,RL8AGCP)
            RUNFF_EXCESS = RUNFF_EXCESS + AMAX1(ABS(DMNDAGCFS)/2.0
     $      -RL8AGCP ,0.0)
            IAGFLO(IAG(IL8AGBSN),1) = NINT(RUNFF_EXCESS)
         ENDIF
      ENDIF
      IF (DMNDAGL8 .GT. 0.00001) THEN
         call match('L8CP  ',1,6,struc_name_meas,nflpts,6,imatch)
         if (imatch .gt. 0) THEN
             qu(il8) = qu(il8) - DMNDAGL8
         endif
      ENDIF

C  For each basin:

C  Each individual cell in the basin can have either excess water or
supplemental demand.

C  If the accumulated excess water in the basin exceeds the accumulated
supplemental demand
C  in the basin, the accumulated excess water meets the accumulated supplemental
demand and
C  the accumulated supplemental demand gets added to the dmnd_met_by_rain.  The
remainder
C  becomes the excess water for the basin.

C  If the accumulated supplemental demand in the basin exceeds the accumulated
excess water
C  in the basin, the accumulated excess water meets part of the accumulated
supplemental
C  demand and the accumulated excess water gets added to the dmnd_met_by_rain.
The
C  remainder becomes the supplemental demand for the basin.

      IF (runmode .eq. 'SIMUL') THEN
         DO 80 IA = 1,NEAABSN-1
            dmnd_met_by_rain = dmnd_met_by_rain + amin1(tdmnd(ia) ,
     $      trunoff(ia)) / 43560.
   80    CONTINUE
C
C   CALCULATE DEMANDS FOR S4 AND S236 BASINS
C
c
c calculate demand for s4 basin
c
           DMDS4 = dmndro(5,2,itypern,jday)
c
c         IF (ifix(total_dmnd_for_298d_grid) .GT. 0) THEN
c            DMDS4 = (TAREAS4) / total_area_298d_grid
c     $      *total_dmnd_for_298d_grid
c         ELSE
```

```
c            DMDS4 = 0.0
c         ENDIF
c
c calculate demand for s-236 basin
c
         DMDS236 = TAREAS236 / total_area_298d_grid
     $   *total_dmnd_for_298d_grid
         DMDS236   = DMDS236 / 43560.

         if (dmds236 .lt. 0.0) then
            s236lok = amin1(abs(dmds236),255*1.9835)
            if (istaopt(8) .eq. 0) then
              dmds236 = - s236lok
            endif
         endif

      ENDIF
      DMNDAGL8 = DMNDAGL8 / 43560.
      DMDSUGH = DMDSUGH / 43560.
      agp(il8bsn-6) = DMNDAGL8*43560.
c
c  excess water within basin meets demands at areas of deficit
c   within basin
c
      IF (runmode .eq. 'SIMUL') THEN
         pot_del_to_eaa_basin(il8bsn-6,indx_lok_del) =
     $     amax1(agp(il8bsn-6),0.0)/86400.
         pot_del_to_eaa_basin(4,indx_lok_del) = amax1(agp(4),0.0)/86400.
c        print *,trunoff(1)/86400.,trunoff(2)/86400.,trunoff(3)/86400.,
c    $      tdmnd(1)/86400.,tdmnd(2)/86400.,agp(is8bsn-6)/86400.,
c    $      agp(is7bsn-6)/86400.,agp(3)/86400.,'   TDMND'
         DO i = maxy,42,-1
            DO node = isum(i) + 1,isum(i+1)
               IF (IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 9) THEN
                  ITYP = LUTYP(NODE)
C
                  K = ETK(ITYP,JDAY)
                  IF (LUTYP(NODE) .EQ. 7) THEN
                     KC = K * ADJCFF(MONTH)
                  ELSEIF(LUTYP(NODE) .EQ. 8) THEN
                     KC = K * ADJCFF(MONTH)
                  ELSEIF(LUTYP(NODE) .EQ. 9) THEN
                     KC = K * ADJCFF(MONTH)
                  ELSE
                     KC = K
                  ENDIF
C
                  ETDM = KC * potet(ietzon(node))/12.
                  ETMX = ETDM
                  depth_below_min = amax1(SOLCRT(NODE,JDAY)-SOLMX(NODE)
     $             ,0.0)
                  dph = depth_below_min + rem_gwdpth(node)
                  depth_of_need = amin1(dph,amax1(etmx-rain(node)/12.
     $            ,0.0))
c
                  IF (depth_of_need .GT. 0.0000001) THEN
                     depth_of_excess = trunoff(ibsn(node)
```

```
     $               -6)/(gdar*(1.0-total_small_afact(node)
     $               -carfact(node)))
                     SOLMX(NODE) = SOLMX(NODE) + amin1(depth_below_min,
     $               depth_of_need,depth_of_excess)
                     excess_water_to_cell(NODE) = excess_water_to_cell
     $               (NODE) + amin1(depth_below_min,depth_of_need,
     $               depth_of_excess)
                     trunoff(ibsn(node)-6) = trunoff(ibsn(node)-6)
     $               - amin1(depth_of_need,depth_below_min
     $               ,depth_of_excess)*GDAR*(1.0-total_small_afact(node)
     $               -carfact(node))
                     perc_to_wt = amin1(depth_of_need,
     $               depth_of_excess) - depth_below_min
                     perc_to_wt = amax1(perc_to_wt,0.0)
                     trunoff(ibsn(node)-6) = trunoff(ibsn(node)-6)
     $               - perc_to_wt*GDAR*(1.0-total_small_afact(node)
     $               - carfact(node))
                     excess_water_to_cell(NODE) = excess_water_to_cell
     $               (NODE) + perc_to_wt
                     trunoff(ibsn(node)-6) = amax1(trunoff(ibsn(node)-6)
     $                ,0.0)
                     RCHG(NODE) = RCHG(NODE) + perc_to_wt
                     rem_gwdpth(node) = rem_gwdpth(node) - perc_to_wt
                  ENDIF
               ENDIF
            ENDDO
         ENDDO
C
c
C  COMPUTE DEMANDS IN MAJOR EAA BASINS TO BE MET
C
         QWS1 = agp(is8bsn-6)/86400.
         QWS2 = agp(is7bsn-6)/86400.
         QWS3 = agp(is5absn-6)/86400.
         sum_eaa_dmnd_ssm(is8bsn-6) = sum_eaa_dmnd_ssm
     $           (is8bsn-6) + amax1(QWS1,0.0)
         if (QWS1 .LT. -0.00001) sum_eaa_dmnd_ssm(is8bsn-6) = 0.0
         sum_eaa_dmnd_ssm(is7bsn-6) = sum_eaa_dmnd_ssm
     $           (is7bsn-6) + amax1(QWS2,0.0)
         if (QWS2 .LT. -0.00001) sum_eaa_dmnd_ssm(is7bsn-6) = 0.0
         sum_eaa_dmnd_ssm(is5absn-6) = sum_eaa_dmnd_ssm
     $           (is5absn-6) + amax1(QWS3,0.0)
         if (QWS3 .LT. -0.00001) sum_eaa_dmnd_ssm(is5absn-6) = 0.0
         sum_alocnnr = sum_alocnnr + amax1(alocnnr,0.0)/86400.
         if (alocnnr/86400. .LT. -0.00001) sum_alocnnr = 0.0
         dmnd_sugh_orig = amax1(dmdsugh,0.0)
c
         iday_of_week = idofw(month,iday,ifyr+iyear-1)
         alocnnr_potdel(indx_lok_del) = 0.0
         dmnd_nnr_orig  = 0.0
         dmnd_nnrhil_orig = 0.0
         pot_del_to_eaa_basin(is8bsn-6,indx_lok_del) = 0.0
         pot_del_to_eaa_basin(is7bsn-6,indx_lok_del) = 0.0
         pot_del_to_eaa_basin(is5absn-6,indx_lok_del) = 0.0
         DO IW = 1,n_days_week_del_eaa(indx_lok_del)
            IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx
     $     (iw,indx_lok_del)) THEN
```

```
                pot_del_to_eaa_basin(is8bsn-6,indx_lok_del) =
     $          sum_eaa_dmnd_ssm(is8bsn-6)
                pot_del_to_eaa_basin(is7bsn-6,indx_lok_del) =
     $          sum_eaa_dmnd_ssm(is7bsn-6)
                pot_del_to_eaa_basin(is5absn-6,indx_lok_del) =
     $          sum_eaa_dmnd_ssm(is5absn-6)
                alocnnr_potdel(indx_lok_del) = sum_alocnnr
                dmnd_nnr_orig  = alocnnr_potdel(indx_lok_del)
                dmnd_nnrhil_orig = pot_del_to_eaa_basin(is7bsn-6
     $          ,indx_lok_del)
c
c  add supplemental volume required to meet crop requirements to soil moisture
c  during the days when deliveries from LOK are made
c
                IF (n_days_week_del_eaa(indx_lok_del) .eq. 7) THEN
c  deliveries are made daily
                do ibb = 1,3
                  vol_of_deliv(ibb) = pot_del_to_eaa_basin(ibb
     $             ,indx_lok_del)
                enddo
                DO  I = MAXY,42,-1
                   DO NODE = ISUM(I)+1,ISUM(I+1)
                      iar = ibsn(node)-6
                      IF (IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 9) THEN
                        IF (land_use_type(LUTYP(NODE)) .NE. 'WETLAND')
     $                  THEN
                           ITYP = LUTYP(NODE)
C
                           K = ETK(ITYP,JDAY)
                           IF (LUTYP(NODE) .EQ. 7) THEN
                             KC = K * ADJCFF(MONTH)
                           ELSEIF(LUTYP(NODE) .EQ. 8) THEN
                             KC = K * ADJCFF(MONTH)
                           ELSEIF(LUTYP(NODE) .EQ. 9) THEN
                             KC = K * ADJCFF(MONTH)
                           ELSE
                             KC = K
                           ENDIF
C
                           ETDM = KC * potet(ietzon(node))/12.
                           ETMX = ETDM
                           depth_below_min =amax1(SOLCRT(NODE,JDAY)
     $                     -SOLMX(NODE),0.0)
                           dph = depth_below_min + rem_gwdpth(node)
                           depth_of_need =amin1(dph,amax1(etmx-rain(node)
     $                     /12.-excess_water_to_cell(node),0.0))
c                          if (node.eq.nodebf) print *,depth_of_need
c    $                     ,dph,depth_below_min,'  BBBB ',month,iday,node
c    $                     ,solmx(node),ETMX,KC,potet(ietzon(node))
c    $                     ,vol_of_deliv(ibsn(node)-6)*1.9835
                           IF (depth_of_need .gt. 0.0000001) THEN
                             dpth_of_deliv = vol_of_deliv
     $                       (ibsn(node)-6)*86400./(gdar*(1.0 -
     $                       total_small_afact(node)-carfact(node)))
                             solmx(node)=solmx(node)+amin1(dpth_of_deliv
     $                       ,depth_below_min,depth_of_need)
                             vol_of_deliv(ibsn(node)-6) = vol_of_deliv
```

```
     $                      (ibsn(node)-6) - amin1(dpth_of_deliv
     $                      ,depth_below_min,depth_of_need)*GDAR/86400.
     $                      *(1.0-total_small_afact(node)-carfact(node))
                            perc_to_wt = amax1(amin1(depth_of_need
     $                      ,dpth_of_deliv)-depth_below_min,0.0)
                            vol_of_deliv(ibsn(node)-6) = vol_of_deliv
     $                      (ibsn(node)-6) - perc_to_wt*GDAR/86400.
     $                      * (1.0 -total_small_afact(node)
     $                      -carfact(node))
                            rchg(node) = rchg(node) + perc_to_wt
                            rem_gwdpth(node) = rem_gwdpth(node) -
     $                      perc_to_wt
c                      IF (node.eq.nodebf)
c    $                  print *,rchg(node),perc_to_wt,depth_of_need
c    $                  ,dpth_of_deliv,'  QQQQ  '
                            ENDIF
                            IF (solmx(node) .gt. solmdph(node)) then
                              POND(node) =AMAX1(SOLMX(node)-SOLMDPH(node)
     $                        ,0.0)
                              SOLMX(node) = solmdph(node)
                            ENDIF
                          ENDIF
                        ENDIF
                     ENDDO
                  ENDDO
c                 print *,vol_of_deliv(1),vol_of_deliv(2),vol_of_deliv(3)
c    $             ,'  VOL OF DELIV  ',pot_del_to_eaa_basin(1,indx_lok_del)
c    $             ,pot_del_to_eaa_basin(2,indx_lok_del)
c    $             ,pot_del_to_eaa_basin(3,indx_lok_del),trunoff(1)/86400.
c    $             ,trunoff(2)/86400.,trunoff(3)/86400.
                  ELSE
c  deliveries are NOT made daily

c
c  add supplemental volume required to meet crop requirements to soil moisture
c  during the days when deliveries from LOK are made
c
                  DO  I = MAXY,42,-1
                     DO NODE = ISUM(I)+1,ISUM(I+1)
                        iar = ibsn(node)-6
                        IF (IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 9) THEN
                          IF (land_use_type(LUTYP(NODE)) .NE. 'WETLAND')
     $                    THEN
                            solmx(node)=solmx(node)+pot_del_to_eaa_basin
     $                      (ibsn(node)-6,indx_lok_del)*86400./
     $                      (NAGNODE(IBSN(NODE)-6)*GDAR*(1.0-carfact(node)
     $                      -total_small_afact(node)))
                            IF (solmx(node) .gt. solmdph(node)) then
                               POND(node) =AMAX1(SOLMX(node)-SOLMDPH(node)
     $                         ,0.0)
                               SOLMX(node) = solmdph(node)
                            ENDIF
                          ENDIF
                        ENDIF
                     ENDDO
                  ENDDO
                  ENDIF
```

```
          ENDIF
        ENDDO
c
c calculate available storage in eaa for excess supply of water from
c Lake Okeechobee when supply-side management is in effect
c
        DO  I = MAXY,42,-1
           DO NODE = ISUM(I)+1,ISUM(I+1)
              IF ((IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 10)
     $          .OR. (IBSN(NODE) .EQ. 40 .OR. IBSN(NODE)
     $          .EQ. 42)) THEN
                iar = ibsn(node)-6
                IF (lp(22).and. land_use_type(lutyp(node)) .eq.
     $          'AGRICUL') THEN
                   avail_stor(node) = amax1(solcrt(node,jday)
     $             - solmx(node),0.0)*gdar*(1.0-carfact(node)
     $             - total_small_afact(node))/43560.
                   IF (ibsn(node) .ne.  40.and. ibsn(node) .ne. 42)
     $             THEN
                      sum_avail_stor(iar) = sum_avail_stor(iar) +
     $                avail_stor(node)
                   ELSE
                      IF (ibsn(node) .eq. 40) THEN
                         sum_avail_l8ag = sum_avail_l8ag
     $                   + avail_stor(node)
                      ELSE
                         sum_avail_sugh = sum_avail_sugh
     $                   + avail_stor(node)
                      ENDIF
                   ENDIF
                   sum_avail_stor_eaa = sum_avail_stor_eaa +
     $             avail_stor(node)
                ENDIF
              ENDIF
           ENDDO
        ENDDO
c
c     calculate total supplemental demands for EAA Basins
c
        SUPP_EAA_DMND = 0.0
        do ia = 1,neaabsn-1
           SUPP_EAA_DMND = SUPP_EAA_DMND + amax1(agp(ia),0.0)/86400.
     $       *1.9835
        enddo
c must limit dmd_met_by_rain due to floating point inaccuracy in cumulative sum.
        dmnd_met_by_rain = amin1(dmnd_met_by_rain,(total_eaa_dmnd -
     $      SUPP_EAA_DMND))
        dmnd_met_by_local_storage = total_eaa_dmnd - SUPP_EAA_DMND -
     $      dmnd_met_by_rain
      ENDIF
c     dmnd_nnr_orig = amax1(alocnnr,0.0)/86400.
c     dmnd_nnrhil_orig = amax1(agp(is7bsn-6),0.0)/86400.
C
      RETURN
      END
```

agdata.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: wwilcox $
C     $Id: agdata.inc,v 1.5 2005/02/10 15:54:02 wwilcox Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/agdata.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------
      common /AGDATA/ AGP(max_n_eaa_condt),NAGPTS,IAGFLO(45,2),
     &NAGNODE(max_n_eaa_condt),BAGP(max_n_eaa_condt),
     &QOUTFLW(max_n_res,max_nostr_res),no_canals_ssm,
     &QINFLW(max_n_res,max_nostr_res),icanal_ssm(max_ncnls),
     &NODEPW(max_n_res,max_nostr_res,max_n_cells_rec),NODEWA(max_n_res),
     &neaacnl,NSINDX(max_n_eaa_condt),PCFS(max_n_eaa_condt),
     &PCFWS(max_n_eaa_condt),FLRNRO(max_n_eaa_condt),
     &FLWSRO(max_n_eaa_condt),ALOCNNR,SOLMX(max_ncells),
     &SOLCRT(max_ncells,366),ADJCFF(12),PCTRUNF(max_n_eaa_condt),
     &NEAABSN,NODEHW(max_n_eaa_condt),MINYNNR,MAXYNNR,
     &IXMXNNR(max_n_rows_nnrb),IXMNNNR(max_n_rows_nnrb),TOTMIAMI,
     &SOLCRNF(max_ncells,366),IAGFLL8,DMNDAGL8,NAGNODEL8,IL8AGBSN,
     &RL8AGRFCAP,RMCAGRFCAP,DMNDAGMC,DMDSUGH,ISUGBSN,NAGNODESG,
     &sum_avail_stor(max_n_eaa_condt),avail_stor(max_ncells),
     &runoff_thres(max_n_eaa_condt,max_n_runff_thres),
     &pct_daily_runoff(max_n_eaa_condt,max_n_runff_thres),
     &nthresholds(max_n_eaa_condt),ieaa_cnl_tw(max_n_eaa_condt),
     &pmp_int(max_n_eaa_condt),stor_eaa_adj(max_n_eaa_condt),
     &sum_avail_stor_eaa,sum_avail_sugh,pmp_int_new(max_n_eaa_condt),
     &ires_opt_hill,dmnd_nnr_orig,iconv_use_inj(max_n_eaa_condt),
     &dmnd_nnrhil_orig,area_298d,eaa_basin_area(max_n_eaa_condt),
     &ratio_runoff_bmp_298d,nnodes_wetl(max_n_eaa_condt),
     &emerg_bypass_sta34,flow_to_s7_sta2byp,flow_to_sta2_frm_eaa,
     &reduced_needs(max_ncnls),vol_rec_from_c51asr,vol_rec_from_hlsbasr,
     &vol_rec_from_nnrcasr,vol_rec_from_l31asr,vol_inj_to_c51asr,
     &vol_inj_to_hlsbasr,vol_inj_to_nnrcasr,vol_inj_to_l31asr,
     &frac_area_eaa_basin(max_n_eaa_condt),is39hw,is39tw,is39tw_opt,
     &is39hw_opt,flow_to_eaa_res_rnff(max_n_eaa_condt),
     &flow_to_eaa_res_reg(max_n_eaa_condt,max_n_reg_rel_basin),
```

```
     &TOTMIAMI_1,total_env_ws_from_eaares,emerg_bypass_s7,
     &emerg_bypass_s8,runoff_to_lec_s8,runoff_to_lec_s7,
     &runoff_to_lec_s6,frac_area_eaa_bmp,sum_avail_l8ag,
     &frac_c139basin_sta5,frac_c139basin_sta6,ic139_rnff_fix_split_opt,
     &NNODESSG_WETL,idest_eaacnl_298d(max_n_eaa_condt)
      real nagnode,NAGNODEL8,NAGNODESG,NNODESSG_WETL
C-----------------
```

alloc_to_eaa.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: alloc_to_eaa.F,v 1.9 2005/06/10 13:28:59 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/alloc_to_eaa.F,v $
C   $Name: rel-5-5 $
C
C   THIS ROUTINE ROUTES THE VOLUME OF WATER DETERMINED BY
C   SUPPLY-SIDE MANAGEMENT PLAN THAT IS IN EXCESS OF THE DEMAND
C   IN LOSA TO CELLS WHERE THE SOIL MOISTURE IS LESS THAN THAN THE
C   MISTURE LEVEL THAT TRIGGERS SUPPLEMENTAL RELEASES FROM OUTSIDE
C   SOURCE.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE alloc_to_eaa (supply_excess,total_small_afact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
c
      DIMENSION  total_small_afact(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (sum_avail_stor_eaa .GT. 0.0) THEN
         ratio_supply_excess = amin1(supply_excess/sum_avail_stor_eaa ,
     $   1.0)
      ELSE
         ratio_supply_excess = 0.0
      ENDIF
c
c
```

```
      DO jb = 1,neaabsn
         IF (agp(jb) .GT. 0.0) THEN
c
            IF (jb .EQ. is8bsn-6) THEN
               add_sugh = sum_avail_sugh
            ELSE
               add_sugh = 0.0
            ENDIF
c
            agp(jb) = agp(jb) + ratio_supply_excess * (sum_avail_stor
     $      (jb) + add_sugh) * 43560.
         ENDIF
      END DO
      dmndagl8 = dmndagl8 + ratio_supply_excess * sum_avail_l8ag
c
c  route excess allocation according to SSM when SSM is in effect to grid
c  cells in EAA that need the water
c
      DO 20 I = MAXY,42,-1
         DO 10 J = ISUM(I) + 1,ISUM(I+1)
            IF (runmode .EQ. 'SIMUL' .OR. IBSN(J) .EQ. 10 .OR.
     $      IBSN(J) .EQ. 40) THEN
               IF ((IBSN(J).LT .7 .OR. IBSN(J) .GT. 10) .AND.
     $         IBSN(J) .NE. 40.AND. IBSN(J) .NE. 42) GO TO 10
c
               IF (land_use_type(lutyp(j)) .EQ. 'AGRICUL') THEN
                  IF (ibsn(j) .EQ. 40) THEN
                     solmx(j) = solmx(j) + ratio_supply_excess *
     $               avail_stor(j) * 43560./(gdar*(1.0-carfact(j)
     $               -total_small_afact(j)))
                  ELSE
                     IF (ibsn(j) .EQ. 42) THEN
                        IF (agp (is8bsn-6) .GE. 0.0) THEN
                           solmx(j) = solmx(j) + ratio_supply_excess *
     $                     avail_stor(j) * 43560./(gdar*(1.0-carfact(j)
     $                     -total_small_afact(j)))
                        ENDIF
                     ELSE
                        IF (agp (ibsn(j)-6) .GE. 0.0) THEN
                           solmx(j) = solmx(j) + ratio_supply_excess *
     $                     avail_stor(j) * 43560./(gdar*(1.0-carfact(j)
     $                     -total_small_afact(j)))
                        ENDIF
                     ENDIF
                  ENDIF
               ENDIF
            ENDIF
   10    CONTINUE
   20 CONTINUE
      RETURN
      END
```

annual_init.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: annual_init.F,v 1.11 2005/06/10 13:28:59 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/annual_init.F,v $
C   $Name: rel-5-5 $
C
C   THIS ROUTINE INITIALIZES VARIABLES AT THE BEGINNING OF EACH
C   YEAR OF SIMULATION
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE annual_init (lhls, iyear, runmode, ifyr)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'ovfllim.inc'

      CHARACTER runmode*5

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C Initialize Over.AND.Flow variables

      DO it = 1,2
         DO node = 1,max_ncells
            ndays_ovlflo_vol_limit(node,iyear,it) = 0
            excess_vol_ovlflo(node,iyear,it) = 0.0
            vol_ovlflo(node,iyear,it) = 0.0
         END DO
      END DO
C
      RETURN
      END
```

asr1.inc

```
c Copyright, 1995, South Florida Water Management District
c
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: asr1.inc,v 1.3 2003/11/17 19:41:27 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr1.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------------------------

C PARAMETER MAX VARIABLES
C    max_n_asr = maximum number of ASR's for the whole domain
C    max_asr_types = max number of ASR types differentiated by ASR
C                    source and destination. Eg. src/dest could be
C                    canal, surface reservoir, eaa ro, eaa dmd, etc.
C    max_src_dest_pts_for_any_asr = max src/destination points FOR any ASR
C    max_asr_for_any_src_or_dest = max number of ASR's associated with any
C                                  source or dest like a cnl, reservoir, etc
C    max_nch = max number of canals. since number of canals is greater than
C              number of reservoirs, or num of eaa basins, max_nch is used
C              as the dimension upper limit for few other variables
C
      integer nasr_from_cnl(max_nch),nasr_to_cnl(max_nch),
     &   nasr_from_res(max_nres),nasr_to_res(max_nres),
     &   nasr_from_eaa(max_nres),nasr_to_eaa(max_nres),
     &   asrnum(max_asr_types,max_ncells,max_asr_for_any_src_or_dest,2),
     &   nnodes_res_to_asr(max_nres,5),icol_asr(max_nres,5,5),
     &   irow_asr(max_nres,5,5),no_ds_cnl_ws_asr(max_nch,5),
     &   icnl_no_ws_asr(max_nch,5,5),nnodes_res_from_asr(max_nres,5),
     &   icol_asr_rec(max_nres,5,5),irow_asr_rec(max_nres,5,5),
     &   nasr_to_lok(5),nasr_from_lok(5),nasr_to_cell(max_ncells)
      real frac_cap_src(max_n_asr,max_nres),
     &     cnl_trig_inj_to_asr(max_nch,5)

      real trig_res2asr(max_nres),trig_resfasr(max_nres,2)
```

```
      character asr_name(max_nres,max_n_asr)*6
     & ,asr_name_rec(max_nres,max_n_asr)*6
      common /asr1/ asrnum,nasr_from_cnl,nasr_to_cnl,
     &    nasr_from_res,nasr_to_res,nasr_from_eaa,
     &    nasr_to_eaa,trig_res2asr,frac_cap_src,
     &    nnodes_res_to_asr,icol_asr,irow_asr,
     &    asr_name,no_ds_cnl_ws_asr,icnl_no_ws_asr,
     &    cnl_trig_inj_to_asr,trig_resfasr,
     &    icol_asr_rec,irow_asr_rec,nnodes_res_from_asr,
     &    asr_name_rec,nasr_to_lok,nasr_from_lok,
     &    nasr_to_cell
C
C-----------------------------------------------------------------------
```

asr2.inc

```
c Copyright, 1995, South Florida Water Management District
c
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: asr2.inc,v 1.2 2003/07/03 20:07:54 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr2.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------------------------

C PARAMETER MAX VARIABLES
C    max_n_asr = maximum number of ASR's for the whole domain
C    max_asr_types = max number of ASR types differentiated by ASR
C                    source and destination. Eg. src/dest could be
C                    canal, surface reservoir, eaa ro, eaa dmd, etc.
C    max_src_dest_pts_for_any_asr = max src/destination points FOR any ASR
C    max_asr_for_any_src_or_dest = max number of ASR's associated with any
C                                  source or dest like a cnl, reservoir, etc

      real asr_incap(max_n_asr), asr_outcap(max_n_asr),
     &     asreffic(max_n_asr), asrvol(max_n_asr), asrloss(max_n_asr),
     &     asr_inj_vol(max_n_asr), asr_rec_vol(max_n_asr)

C     integer asr_inj_recv_flag

      integer cal_asrnum,stl_asrnum,cal_asr_flag,stl_asr_flag

      common /asr2/ asr_incap,asr_outcap,asreffic,ntotasr,asrvol,asrloss
     &             , asr_inj_vol,asr_rec_vol,
     &               cal_asrnum,stl_asrnum,cal_asr_flag,stl_asr_flag
C
C-------------------------------------------------------------------
```

asr3.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: asr3.inc,v 1.4 2004/03/08 21:41:48 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr3.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------------------------

C PARAMETER MAX VARIABLES
C    max_n_asr = maximum number of ASR's for the whole domain
C    max_asr_types = max number of ASR types differentiated by ASR
C                    source and destination. Eg. src/dest could be
C                    canal, surface reservoir, eaa ro, eaa dmd, etc.
C    max_src_dest_pts_for_any_asr = max src/destination points FOR any ASR
C    max_asr_for_any_src_or_dest = max number of ASR's associated with any
C                                  source or dest like a cnl, reservoir, etc

      integer ncnl_src_to_asr(max_n_asr),nres_src_to_asr(max_n_asr),
     &     neaa_src_to_asr(max_n_asr), ncnl_dest_from_asr(max_n_asr),
     &     nres_dest_from_asr(max_n_asr),neaa_dest_from_asr(max_n_asr),
     &     cnl_src_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     res_src_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     eaa_src_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     cnl_dest_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     res_dest_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     eaa_dest_num(max_n_asr,max_src_dest_pts_for_any_asr),
     &     no_cnl_ws(max_n_asr,max_src_dest_pts_for_any_asr),
     &     icnl_no_ws(max_n_asr,max_src_dest_pts_for_any_asr,5),
     &     nlok_src_to_asr(max_n_asr),nlok_dest_from_asr(max_n_asr),
     &     ncells_dest_from_asr(max_n_asr)

        real trig_asr(max_n_asr,max_src_dest_pts_for_any_asr),
     &     rem_incap(max_n_asr),total_asr_inj(max_n_asr),
     &     total_asr_rec(max_n_asr),rem_asr_outcap(max_n_asr),
     &     rmin_bubble_sze_add_rec(max_n_asr),rec_asr_outcap(max_n_asr),
     &     asr_cap_f_pws(max_n_asr),asr_bubble_vol_lim(max_n_asr)
```

```
      common /asr3/ trig_asr,ncnl_src_to_asr,nres_src_to_asr
     &  ,neaa_src_to_asr,ncnl_dest_from_asr,nres_dest_from_asr
     &  ,neaa_dest_from_asr,cnl_src_num,res_src_num,eaa_src_num
     &  ,cnl_dest_num,res_dest_num,eaa_dest_num,rem_incap
     &  ,total_asr_inj,no_cnl_ws,icnl_no_ws,total_asr_rec
     &  ,rem_asr_outcap,nlok_src_to_asr,nlok_dest_from_asr
     &  ,rmin_bubble_sze_add_rec,rec_asr_outcap
     &  ,ncells_dest_from_asr,asr_cap_f_pws,asr_bubble_vol_lim
C
C-----------------------------------------------------------------------
```

asr.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: asr.F,v 1.8 2005/06/10 13:28:59 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO DO WATER BUDGET ACCOUNTING FOR THE PARTICULAR AQUIFER
C     STORAGE AND RECOVERY (ASR) SYSTEM
C
C   VARIABLES:
C    asrid     UNIQUE NUMBER ASSOCIATED WITH EACH ASR
C    ASR_INJ_RECV_FLAG    FLAG THAT INDICATES
C                  1 = perform injection into the particular ASR called
C                  2 = perform recovery from the particular ASR called
C    VOL_IN       Volume (acft) that needs to be injected/recovered
C    BAL_VOL      Volume (acft) that could not be injected/recovered
C    ASRVOL(asrid) - available volume in the asr bubble that could be
C                    fully used to meet demands.
C    ASRID        an index running from 1 to total number of asr's
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE asr (asrid, asr_inj_recv_flag, vol_in, bal_vol,
     & res_src_no, tot_cnl_dmd_asr)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'stas.inc'

      INTEGER  asr_inj_recv_flag, asrid, res_src_no
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2


C  IF injection phase for the asr, increment asr volume by amount that
C  can be injected (limited by asr capacity and vol available for injection)
C  asr capacity is in mgd..convert to acft
C  Volume incremented in the bubble during each injection (which is fully
availab
C  to meet demands) is = vol injected*asreffic.

      IF (asr_inj_recv_flag .EQ. 1) THEN
         IF (res_src_no .EQ. 0) THEN
            frac_cap_src_in = 1.0
         ELSE
            frac_cap_src_in = frac_cap_src(asrid,res_src_no)
         ENDIF
         asr_max_in = rem_incap(asrid)*3.0689*frac_cap_src_in
         rmax_avail_vol = max(asr_bubble_vol_lim(asrid)
     $    - asrvol(asrid),0.0)
         asr_inj_vol(asrid) = amin1(vol_in,asr_max_in,rmax_avail_vol)
         total_asr_inj(asrid) = total_asr_inj(asrid) + asr_inj_vol
     $   (asrid)
         asrvol(asrid) = asrvol(asrid) + asr_inj_vol(asrid)* asreffic
     $   (asrid)
         asrloss(asrid) = asr_inj_vol(asrid)*(1.0 - asreffic(asrid))
         rem_incap(asrid) = asr_incap(asrid)-total_asr_inj(asrid)/
     $   (3.0689)
C
C compute volume that could not be injected
C
         bal_vol = vol_in - asr_inj_vol(asrid)
      ENDIF
C-----------------------------------------------------------------------
C IF recovery phase from the asr, decrement asr volume by amount that
C can be recovered (limited by asr capacity and available vol)
C asr capacity is in mgd..convert to acft
C-----------------------------------------------------------------------
      IF (asr_inj_recv_flag .EQ. 2) THEN
         available_vol = amin1(asrvol(asrid),rem_asr_outcap(asrid) *
     $   3.0689)
         available_vol = available_vol - tot_cnl_dmd_asr*1.9835
         available_vol = amax1(available_vol,0.0)
         asr_rec_vol(asrid)   = amin1(vol_in,available_vol)
         total_asr_rec(asrid) = total_asr_rec(asrid) + asr_rec_vol
     $   (asrid)
         rem_asr_outcap(asrid) = rec_asr_outcap(asrid) - total_asr_rec
     $   (asrid) / 3.0689
         asrvol(asrid) = asrvol(asrid) - asr_rec_vol(asrid)
C compute volume that could not be recovered
         bal_vol = vol_in - asr_rec_vol(asrid)
      ENDIF

      RETURN
      END
```

asr_input.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: asr_input.F,v 1.14 2004/03/08 21:41:47 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr_input.F,v $
C   $Name: rel-5-5 $
C
C
C  In this subroutine, READ in basic parameters for each ASR Bubble.
C  Each ASR has an efficiency factor defined as
C     eta = (volume recoverable)/(vol injected), and a capacity
C           (separate for injection and recovery).
C  Any ASR can have multiple sources in the injection phase
C  (e.g. canals, reservoirs, eaa ro, etc), and during the recovery phase,
C  multiple destinations.
C   Thus, for each ASR, the foll. have to be defined
C    1) number of canal sources, IF any, FROM which the asr takes water for
C          injection(ncnl_src_to_asr)
C    2) number of reservoirs srcs, IF any, FROM which the asr takes water
C          for injection(nres_src_to_asr)
C    3) number of eaa ro sources, IF any, FROM which the asr takes water
C          for injection(neaa_src_to_asr)
C   Also needed, for each ASR, are the following
C    1) number of canal destination, IF any, TO which the asr delivers
C          during recovery(ncnl_dest_from_asr)
C    2) number of reservoirs dest, IF any, TO which the asr delivers
C          during recovery(nres_dest_from_asr)
C    3) number of eaa dmd dest, IF any, TO which the asr delivers
C          during recovery(neaa_dest_from_asr)
C
C  nasr_from_cnl(cnl_num) = is the number of ASR's associated with a particular
C                           particular canal "cnl_num" serving as its source
C  nasr_to_cnl(cnl_num) = is the number of ASR's associated with a particular
C                         canal "cnl_num" serving as its desination
C  nasr_from_res(res_num) = is the number of ASR's associated with a particular
C                           reservoir "res_num" serving as its source
C  nasr_to_res(res_num) = is the number of ASR's associated with a particular
```

```
C                          reservoir "res_num" serving as its destination
C  nasr_from_eaa(eaa_num) = is the number of ASR's associated with a particular
C                            eaa "eaa_num" serving as its source
C  nasr_to_eaa(eaa_num) = is the number of ASR's associated with a particular
C                          eaa "eaa_num" serving as its destination
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE asrinput (iuse_asr_env1, iuse_asr_env2, charid_asr)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'

      DIMENSION charid_asr(max_n_asr,max_src_dest_pts_for_any_asr),
     & charid_asr_rec(max_n_asr,max_src_dest_pts_for_any_asr),
     & cnl_stg_trig_inj_asr(max_n_asr,max_src_dest_pts_for_any_asr),
     & lake_dest_num(max_n_asr,max_src_dest_pts_for_any_asr),
     & lake_src_num(max_n_asr,max_src_dest_pts_for_any_asr),
     & trig_asr_rec(max_n_asr,max_src_dest_pts_for_any_asr,2),
     & icol_dest_from_asr(max_n_asr,max_src_dest_pts_for_any_asr),
     & irow_dest_from_asr(max_n_asr,max_src_dest_pts_for_any_asr)

      CHARACTER charid_asr*6, charid_asr_rec*6,
     & cnl_dest_name(max_src_dest_pts_for_any_asr)*5,
     & cnl_src_name(max_src_dest_pts_for_any_asr)*5,
     & icnl_no_ws_name(10)*5,
     & res_dest_name(max_src_dest_pts_for_any_asr)*6,
     & res_src_name(max_src_dest_pts_for_any_asr)*6

      INTEGER cnl_num,res_num,eaa_num

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2


C Read the total number of ASR's for the entire model domain
C Also READ Caloosahatchee & St Lucie flags: 1 => asr exists

      READ (17,*) NTOTASR, cal_asr_flag, stl_asr_flag ,iuse_asr_env1 ,
     $iuse_asr_env2


C for each ASR, READ its properties, sources and destinations
C The asr properties should always be defined FIRST for caloosahatchee THEN st l
C (in that order) IF they exist.

      DO 50 I = 1,NTOTASR

C-------------------------------------------------------------------------------
```

```
C make sure NTOTASR <= max_n_asr
C-----------------------------------------------------------------------------
         IF(NTOTASR .gt. max_n_asr) THEN
            WRITE(*,*) 'total # of ASRs is > max_n_asr=',max_n_asr
            WRITE(*,*) 'Increase max_n_asr OR decrease NTOTASR.'
            stop
         ENDIF
C-----------------------------------------------------------------------------
C READ the asr capacites (injection and recovery) in MGD and recovery efficiency
C factor(a 0.6 factor means 60% recovery, 40% loss),also read minimum bubble
C size required for recovery rate can be larger than injection rate.
C-----------------------------------------------------------------------------
         READ (17,10) asr_incap(I),asr_outcap(I),asreffic(I)
     $         ,rmin_bubble_sze_add_rec(i),asr_bubble_vol_lim(I)
c        WRITE(*,1005) asr_incap(i),asr_outcap(i),asreffic(i)

C-----------------------------------------------------------------------------
C for this ASR, define number of canal sources and the canal number for each cnl
C IMPORTANT: for caloos & stlucie asrs, define number of canal sources =0
C-----------------------------------------------------------------------------
         READ (17,'(I5,2x,10(A5,2X,F6.2,2X))') ncnl_src_to_asr(I),
     $   (cnl_src_name(J),cnl_stg_trig_inj_asr(i,j), j=1,ncnl_src_to_asr
     $   (I))
         DO j = 1,ncnl_src_to_asr(I)
            call match(cnl_src_name(J),1,5,cnm,nch+1,5,imatch)
            cnl_src_num(i,j) = imatch
         END DO

C-----------------------------------------------------------------------------
C DO same for reservoir sources. In addition, IF the asr has reservoirs, THEN
C define for each of these reservoirs,a trigger level above which injection to
C its asr's can occur.
C IMPORTANT: for caloos & stlucie asrs, define number of reservoir sources =0
C-----------------------------------------------------------------------------
         READ (17,'(I5,2x,10(A6,2X,F6.2,2X))') nres_src_to_asr(I),
     $   (res_src_name(J),frac_cap_src(I,J),J=1,nres_src_to_asr(I))
         IF(nres_src_to_asr(I) .gt. 0) THEN
            DO j = 1,nres_src_to_asr(I)
               call match(res_src_name(J),1,6,resname,ntotres,6,imatch)
               res_src_num(i,j) = imatch
            END DO
            DO j = 1,nres_src_to_asr(I)
              READ (17,40) charid_asr(i,j),trig_asr(I,J) ,
     $        nnodes_res_to_asr(res_src_num(i,j),j),(icol_asr(
     $        res_src_num(i,j),j,k),irow_asr(res_src_num(i,j)
     $        ,j,k),k=1, nnodes_res_to_asr(res_src_num(i,j)
     $        ,j))
            ENDDO
         ENDIF

C-----------------------------------------------------------------------------
C DO same for eaa ro sources
C IMPORTANT: for caloos & stlucie asrs, define number of eaa ro sources =0
C-----------------------------------------------------------------------------
         READ (17,30) neaa_src_to_asr(I), (eaa_src_num(I,J),J=1,
     $   neaa_src_to_asr(I))
C-----------------------------------------------------------------------------
```

```
C DO same for LOK as direct source
C-----------------------------------------------------------------------------
         READ (17,30) nlok_src_to_asr(I), (lake_src_num(I,J),J=1,
     $   nlok_src_to_asr(I))
C-----------------------------------------------------------------------------
C next for this ASR, define number of canal destinations and the canal number fo
C for caloos & stlucie asr's, define number of canal destinations=0
C-----------------------------------------------------------------------------
         READ (17,'(I5,2x,10(A5,2X))') ncnl_dest_from_asr(I),
     $   (cnl_dest_name(J),J=1,ncnl_dest_from_asr(I))
         DO j = 1,ncnl_dest_from_asr(I)
            call match(cnl_dest_name(j),1,5,cnm,nch+1,5,imatch)
            cnl_dest_num(I,J) = imatch
         END DO
         DO j = 1,ncnl_dest_from_asr(I)
            READ (17,'(I5,2x,10(A5,2X))') no_cnl_ws(i,j) ,
     $      (icnl_no_ws_name(k),k = 1,no_cnl_ws(i,j))
            DO k = 1,no_cnl_ws(i,j)
               call match(icnl_no_ws_name(k),1,5,cnm, nch+1,5,imatch)

               IF (imatch .eq. 0) THEN
                  WRITE (*,20) icnl_no_ws_name(k)
               ENDIF
               icnl_no_ws(i,j,k) = imatch
            END DO
         END DO

C-----------------------------------------------------------------------------
C DO same for reservoir destinations
C for caloos & stlucie asr's, define number of reservoir destinations=0
C-----------------------------------------------------------------------------
         READ (17,'(I5,2x,10(A6,2X))') nres_dest_from_asr(I),
     $   (res_dest_name(J),J=1,nres_dest_from_asr(I))
         DO j= 1,nres_dest_from_asr(I)
            call match(res_dest_name(j),1,6,resname,ntotres,6,imatch)
            res_dest_num(I,J) = imatch
         END DO
         IF (nres_dest_from_asr(I) .gt. 0) then
           DO j = 1,nres_dest_from_asr(I)
             READ (17,160) charid_asr_rec(i,j),trig_asr_rec(I,J,1)
     $       ,trig_asr_rec(I,J,2),nnodes_res_from_asr(res_dest_num(i,j)
     $       ,j),(icol_asr_rec(res_dest_num(I,J),J,K)
     $       ,irow_asr_rec(res_dest_num(I,J),j,k),k=1
     $       ,nnodes_res_from_asr(res_dest_num(I,J),j))
           ENDDO
         ENDIF


C-----------------------------------------------------------------------------
C DO same for eaa dmd destinations
C for caloos & stlucie asr's, define number of eaa dmd destinations=0
C-----------------------------------------------------------------------------
         READ (17,30) neaa_dest_from_asr(I), (eaa_dest_num(I,J),J=1,
     $   neaa_dest_from_asr(I))
C-----------------------------------------------------------------------------
C DO same for LOK as direct destination
C-----------------------------------------------------------------------------
```

```
         READ (17,30) nlok_dest_from_asr(I), (lake_dest_num(I,J),J=1,
     $   nlok_dest_from_asr(I))
C-----------------------------------------------------------------------------
C Do same for grid cell(s) outside reservoirs as destinations
C-----------------------------------------------------------------------------
         READ (17,*) asr_cap_f_pws(I),ncells_dest_from_asr(I),
     $   (icol_dest_from_asr(I,J),irow_dest_from_asr(I,J),J=1,
     $   ncells_dest_from_asr(I))
   50 CONTINUE


C-----------------------------------------------------------------------------
C set the asrnum #'s to establish linkage between an ASR and its
C source and destination points. The asrnum(i,j,k,l) 4D array takes
C care of this. The value of asrnum(i,j,k,l) is nothing but just the
C INTEGER index running from 1 to the total number of asr's. Thus,
C 1 <= value of asrnum(i,j,k,l) <= NTOTASR
C
C In asrnum(I,J,K,L) array
C   I = Source/Destination type
C        = 1 for canal
C        = 2 for surface reservoir
C        = 3 eaa runoff or demand
C        = 4 LOK
C        = 5 Grid Cell
C   J = Source/Destination Number
C        = canal # (if I = 1), lying between 1 and max_nch
C        = reservoir # (if I = 2), lying between 1 and NTOTRES
C        = eaa ro/dmd # (if I = 3), lying between 1 and NEAABSN
C        = 1 (if I = 4)
C        = col number,row number
C       Since, max_nch is largest among max_nch ,NTOTRES,NEAABSN, J <= max_nch
C   K = An index representing the number of ASR's associated with the
C       particular source/destination. 0 <= K <= max_asr_for_any_src_or_dest
C   L = 1 when the asr is being injected
C     = 2 when recovery is being done from the asr
C
C Caloosahatchee & St Lucie asr's are handled seperately (see end)
C-----------------------------------------------------------------------------


      DO 120 I = 1, NTOTASR

C-----------------------------------------------------------------------------
C first all canals which serve as source for this ASR I
C nasr_from_cnl(cnl_num) is the number of ASR's associated with canal "cnl_num"
C-----------------------------------------------------------------------------
         DO 60 J = 1, ncnl_src_to_asr(I)
            cnl_num = cnl_src_num(I,J)
            nasr_from_cnl(cnl_num) = nasr_from_cnl(cnl_num) + 1
            m = nasr_from_cnl(cnl_num)
            asrnum(1,cnl_num,m,1) = i
            cnl_trig_inj_to_asr(cnl_num,m) = cnl_stg_trig_inj_asr(i,j)

   60    CONTINUE
c     WRITE(*,*) 'nasr_from_cnl',(nasr_from_cnl(ii), ii=1,max_nch)
```

```
C-------------------------------------------------------------------------------
C next all canals which serve as destinations for this ASR I
C nasr_to_cnl(cnl_num) gives the number of ASR's associated with canal cnl_num a
C-------------------------------------------------------------------------------
C
         DO 70 J = 1, ncnl_dest_from_asr(I)
            cnl_num = cnl_dest_num(I,J)
            nasr_to_cnl(cnl_num) = nasr_to_cnl(cnl_num) + 1
            m = nasr_to_cnl(cnl_num)
            no_ds_cnl_ws_asr(cnl_num,m) = no_cnl_ws(i,j)
            DO k = 1,no_cnl_ws(i,j)
               icnl_no_ws_asr(cnl_num,m,k) = icnl_no_ws(i,j,k)
            END DO
            asrnum(1,cnl_num,m,2) = I
   70    CONTINUE
c     WRITE(*,*) 'nasr_to_cnl',(nasr_to_cnl(ii), ii=1,max_nch)

C-------------------------------------------------------------------------------
C next all reservoirs which serve as source for this ASR I
C nasr_from_res(res_num) gives the number of ASR's associated with reservor res_
C-------------------------------------------------------------------------------
C
         DO 80 J = 1, nres_src_to_asr(I)
            res_num = res_src_num(I,J)
            trig_res2asr(res_num) = trig_asr(I,J)
            nasr_from_res(res_num) = nasr_from_res(res_num) + 1
            m = nasr_from_res(res_num)
            asr_name(res_num,m) = charid_asr(i,j)
            asrnum(2,res_num,m,1) = I
   80    CONTINUE
c     WRITE(*,*) 'nasr_from_res',(nasr_from_res(k), k=1,10)
c     WRITE(*,*) 'trig ',(trig_res2asr(k), k=1,10)
C-----------------------------------------------------------------------
C next LOK as source
C-----------------------------------------------------------------------
         DO    J = 1, nlok_src_to_asr(I)
            lok_num = lake_src_num(I,J)
            nasr_from_lok(lok_num) = nasr_from_lok(lok_num) + 1
            m = nasr_from_lok(lok_num)
            asrnum(4,J,m,1) = I
         END DO

C-------------------------------------------------------------------------------
C next all reservoirs which serve as destinations for this ASR I
C nasr_to_res(res_num) gives the number of ASR's associated with reservor res_nu
C-------------------------------------------------------------------------------
C
         DO 90 J = 1, nres_dest_from_asr(I)
            res_num = res_dest_num(I,J)
            trig_resfasr(res_num,1) = trig_asr_rec(i,j,1)
            trig_resfasr(res_num,2) = trig_asr_rec(i,j,2)
            nasr_to_res(res_num) = nasr_to_res(res_num) + 1
            m = nasr_to_res(res_num)
            asr_name_rec(res_num,m) = charid_asr_rec(i,j)
            asrnum(2,res_num,m,2) = I
   90    CONTINUE
```

```
c     WRITE(*,*) 'nasr_to_res',(nasr_to_res(ii), ii=1,10)

C-------------------------------------------------------------------------------
C next all eaa ro's which serve as source for this ASR I
C nasr_from_eaa(eaa_num) gives the number of ASR's associated with eaa ro "eaa_n
C-------------------------------------------------------------------------------
         DO 100 J = 1, neaa_src_to_asr(I)
            eaa_num = eaa_src_num(I,J)
            nasr_from_eaa(eaa_num) = nasr_from_eaa(eaa_num) + 1
            m = nasr_from_eaa(eaa_num)
            asrnum(3,eaa_num,m,1) = I
  100    CONTINUE
c     WRITE(*,*) 'nasr_from_eaa',(nasr_from_eaa(ii), ii=1,3)

C-------------------------------------------------------------------------------
C next all eaa dmd's which serve as destinations for this ASR I
C nasr_to_eaa(eaa_num) gives the number of ASR's associated with eaa dmd eaa_num
C-------------------------------------------------------------------------------
         DO 110 J = 1, neaa_dest_from_asr(I)
            eaa_num = eaa_dest_num(I,J)
            nasr_to_eaa(eaa_num) = nasr_to_eaa(eaa_num) + 1
            m = nasr_to_eaa(eaa_num)
            asrnum(3,eaa_num,m,2) = I
  110    CONTINUE
c     WRITE(*,*) 'nasr_to_eaa',(nasr_to_eaa(ii), ii=1,3)
C-----------------------------------------------------------------------
C next LOK as destination
C-----------------------------------------------------------------------
         DO    J = 1, nlok_dest_from_asr(I)
            lok_num = lake_dest_num(I,J)
            nasr_to_lok(lok_num) = nasr_to_lok(lok_num) + 1
            m = nasr_to_lok(lok_num)
            asrnum(4,lok_num,m,2) = I
         END DO
C-----------------------------------------------------------------------
C  next Grid Cell(s) outside Reservoirs as destinations
C-----------------------------------------------------------------------
         DO  J = 1,ncells_dest_from_asr(I)
            icell_num = icol_dest_from_asr(I,J)
     $      - minx(irow_dest_from_asr(I,J)) + 1
     $      + isum(irow_dest_from_asr(I,J))
            nasr_to_cell(icell_num) = nasr_to_cell(icell_num)
     $      + 1
            m = nasr_to_cell(icell_num)
            asrnum(5,icell_num,m,2) = I
         ENDDO
  120 CONTINUE
c
C-----------------------------------------------------------------------
c  input nodal locations for injection from reservoir to asr
c-----------------------------------------------------------------------
c     DO ia = 1,ntotres
c        IF (nasr_from_res(ia) .gt. 0) THEN
c           DO j=1,nasr_from_res(ia)
c              READ (17,*) nnodes_res_to_asr(ia,j) ,(icol_asr(ia,j,k),
c    $         irow_asr(ia,j,k),k=1, nnodes_res_to_asr(ia,j))
c           END DO
```

```
c        ENDIF
C-----------------------------------------------------------------
c  input nodal locations for recovery from asr to reservoir
c-----------------------------------------------------------------
c        IF (nasr_to_res(ia) .gt. 0) THEN
c           DO j=1,nasr_to_res(ia)
c              READ (17,*) nnodes_res_from_asr(ia,j) ,(icol_asr_rec
c    $         (ia,j,k),irow_asr_rec(ia,j,k) ,k=1,nnodes_res_from_asr
c    $         (ia,j))
c           END DO
c        ENDIF
c     END DO
C-------------------------------------------------------------------------------
C now for Caloosahatchee and St Lucie asr's
C Use stl_asrnum as the asrid for injecting/recovering from its asr
C Use cal_asrnum as the asrid for injecting/recovering from its asr
C-------------------------------------------------------------------------------
      stl_asrnum = ntotasr * stl_asr_flag
      cal_asrnum = ntotasr * cal_asr_flag - stl_asr_flag

C-------------------------------------------------------------------------------
C print out ASR properties
C-------------------------------------------------------------------------------
      WRITE(*,*)
      WRITE(*,*) 'NUMBER OF ASRs SIMULATED=', NTOTASR
      WRITE(*,*)
      WRITE(*,*) 'ASR CAPACITIES AND RECOVERY EFFICIENCIES '
      WRITE(*,*) '----------------------------------------'
      DO I = 1,NTOTASR
         WRITE (*,*) 'ASR Number                  =', I
         WRITE (*,*) 'ASR INJECT CAP in MGD       =', asr_incap(I)
         WRITE (*,*) 'ASR RECOVER CAP in MGD      =', asr_outcap(I)
         WRITE (*,*) 'ASR RECOVERY EFFICIENCY in %=', asreffic(I)
         WRITE(*,*)
      END DO
      WRITE(*,*)
      WRITE(*,*) 'ASR SOURCE DETAILS'
      WRITE(*,*) '------------------'
      DO i=1,max_asr_types
         DO j=1,max_nch
            DO k=1,max_asr_for_any_src_or_dest
               IF(asrnum(i,j,k,1).ne.0) THEN
                  IF(i.eq.1) THEN
                     WRITE(*,*) ' ASR Source Type= CANAL'
                     WRITE(*,*) ' Source Canal Number =',j
                     WRITE(*,*) ' Injecting into ASR =',asrnum(i,j,k,1)
                     WRITE(*,*)
                  ELSE IF(i.eq.2) THEN
                     WRITE(*,*) ' ASR Source Type= RESERVOIR'
                     WRITE(*,*) ' Source Reservoir Number =',j
                     WRITE(*,*) ' Source Reservoir Trigger Level==',
     $               trig_res2asr(j)
                     WRITE(*,*) ' Injecting into ASR =',asrnum(i,j,k,1)
                     WRITE(*,*)
                  ELSE IF(i.eq.3) THEN
                     WRITE(*,*) ' ASR Source Type= EAA RO'
                     WRITE(*,*) ' Source Eaa runoff Number =',j
```

```
                  WRITE(*,*) ' Injecting into ASR =',asrnum(i,j,k,1)
                  WRITE(*,*)
               ENDIF
            ENDIF
         END DO
      END DO
   END DO
   WRITE(*,*) 'ASR DESTINATION DETAILS'
   WRITE(*,*) '-----------------------'
   DO i=1,max_asr_types
      DO j=1,max_nch
         DO k=1,max_asr_for_any_src_or_dest
            IF(asrnum(i,j,k,2).ne.0) THEN
               IF(i.eq.1) THEN
                  WRITE(*,*) ' ASR Destination Type= CANAL'
                  WRITE(*,*) ' Destination Canal Number =',j
                  WRITE(*,*) ' Recovery from ASR =',asrnum(i,j,k,2)
                  WRITE(*,*)
               ELSE IF(i.eq.2) THEN
                  WRITE(*,*) ' ASR Destination Type= RESERVOIR'
                  WRITE(*,*) ' Destination Reservoir Number =',j
                  WRITE(*,*) ' Recovery from ASR =',asrnum(i,j,k,2)
                  WRITE(*,*)
               ELSE IF(i.eq.3) THEN
                  WRITE(*,*) ' ASR Destination Type= EAA DMD'
                  WRITE(*,*) ' Destination Eaa DMD Number =',j
                  WRITE(*,*) ' Recovery from ASR =',asrnum(i,j,k,2)
                  WRITE(*,*)
               ENDIF
            ENDIF
         END DO
      END DO
   END DO

C for caloosahatchee & st lucie
   IF (cal_asr_flag.eq.1) WRITE(*,*) ' Caloosahatchee ASR Simulated'
   IF (stl_asr_flag.eq.1) WRITE(*,*) ' St Lucie ASR Simulated'
   WRITE(*,*)

   RETURN
 10 FORMAT (4F10.3,F15.0)
 20 FORMAT(2(/),'UNRECOGNIZED CANAL NAME',2x,A5 ,2x,'FOR WS FROM ASR')

 30 FORMAT (I5,10I5)
 40 FORMAT (2X,A6,F7.1,I3,1x,10(2i3))
160 FORMAT (2X,A6,2F7.1,I3,1x,10(2i3))
   END
```

asr_param.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: asr_param.inc,v 1.4 2003/11/17 19:41:27 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr_param.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------------------------

C PARAMETER MAX VARIABLES
C    max_n_asr = maximum number of ASR's for the whole model domain
C    max_asr_types = max number of ASR types differentiated by ASR
C                    source and destination. Eg. src/dest could be
C                    canal, surface reservoir, eaa ro, eaa dmd, etc.
C    max_src_dest_pts_for_any_asr = max src/destination points FOR any ASR
C    max_asr_for_any_src_or_dest = max number of ASR's associated with
C                               any source or dest. like a cnl, reservoir, etc
C    max_nch = max num of canals
C    max_nres = max num of reservoirs

      parameter (max_n_asr=10, max_asr_types=5,
     &           max_src_dest_pts_for_any_asr=5,
     &           max_asr_for_any_src_or_dest=4,max_nch=220,
     &           max_nres=30)
C
C-----------------------------------------------------------------------
```

asr_to_lec_ws.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: asr_to_lec_ws.F,v 1.12 2005/06/10 13:29:00 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/asr_to_lec_ws.F,v $
C   $Name: rel-5-5 $
C
C
C  THIS SUBROUTINE CALCULATES VOLUMES OF RECOVERY FROM AQUIFER STORAGE
C  AND RETRIEVAL SYSTEMS TO MAINTAIN CANALS IN LEC SERVICE AREAS DURING
C  DRY PERIODS. SOURCES OF WATER FOR RECOVERY FROM ASR WELLS CAN BE
C  EXCESS WATER FROM RESERVOIRS OR CANAL SYSTEMS.
C
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE asr_to_lec_ws (ires_no, flow_from_asr_ws, asr_rec_ws,
     & asr_rec_ws_cnty, asr_rec_vol_wca, wsnde1, wsnde2, wsnde3, wsndes,
     & wsndese, vol_rec_from_cpbasr, icounty_loc_indx, tot_cnl_dmd_asr,
     & ic51, vol_rec_from_acmasr,asr_t_cell_volm)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'resws.inc'
```

```
      DIMENSION asr_rec_vol_wca(max_n_wcas),
     & asr_rec_ws(max_n_serv_area), asr_rec_ws_cnty(max_n_serv_area),
     & flow_from_asr_ws(max_n_res,max_nostr_res),
     & icounty_loc_indx(max_ncnls), ires_no(max_n_res),
     & asr_t_cell_volm(max_ncells)

      CHARACTER char_id_cnl_ws_resasr*5

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   Recovery from ASR wells associated with reservoirs

      DO irs = 1,nres_ws
         DO ib = 1,neaabsn-1
            IF (ires_no(irs) .eq. ieaa_res_asr(ib)) THEN
               GO TO 20
            ENDIF
         END DO
         DO is = 1,noutlets(irs)
            flow_from_asr_ws(irs,is) = 0.0
            js = istrctr_ws(ires_no(irs),is)
            IF (inodopt(ires_no(irs),js,3) .eq. 0) THEN
               icnl_no = ircnl(ires_no(irs),js,1)
               char_id_cnl_ws_resasr = cnm(ircnl(ires_no(irs),js,1))
               IF (char_id_cnl_ws_resasr(1:3) .ne. 'LWD') THEN
                  DO icnl = 1,nasr_to_cnl(icnl_no)
                     IF (nres_src_to_asr(asrnum(1,icnl_no,icnl,2)) .gt.
     $               0) THEN
                        vol_in = dqu(icnl_no) * 1.9835
                        call asr (asrnum(1,icnl_no,icnl,2), 2,vol_in,
     $                  bal_vol,1,tot_cnl_dmd_asr)
                        flow_from_asr_ws(irs,is) = vol_in - bal_vol +
     $                  flow_from_asr_ws(irs,is)
                        qu(icnl_no) = qu(icnl_no) + (vol_in-bal_vol) *
     $                  43560.
                        if (lcnb(icnl_no) .gt. 6.or. lcnb(icnl_no).eq.0)
     $                  then
                          asr_rec_ws(iserv_area_loc(icnl_no))
     $                    = asr_rec_ws(iserv_area_loc(icnl_no))
     $                    + (vol_in-bal_vol)/1.9835
                          asr_rec_ws_cnty(icounty_loc_indx(icnl_no))
     $                    = asr_rec_ws_cnty(icounty_loc_indx(icnl_no))
     $                    + (vol_in-bal_vol)/1.9835
                        else
                          asr_rec_vol_wca(lcnb(icnl_no)) =
     $                    asr_rec_vol_wca(lcnb(icnl_no))
     $                    + (vol_in-bal_vol)/1.9835
                        endif
                        dqu(icnl_no) = bal_vol / 1.9835
                        ICNL_DNS = icnl_no
   10                   ICNL_DNS = iup_canal_ws(ICNL_DNS)
                        IF (ICNL_DNS .LE. NCH) THEN
                           dqu(ICNL_DNS) = dqu(ICNL_DNS) - (vol_in
     $                     -bal_vol)/1.9835
                           dqu(ICNL_DNS) = AMAX1(dqu(ICNL_DNS),0.0)
                           GO TO 10
```

```
                     ENDIF
                  ENDIF
               END DO
            ENDIF
         ENDIF
      END DO
   20 CONTINUE
   END DO
c
c  recovery from asr wells to help public water utilities
c
   if (month .ge. 3 .and. month .le. 5) then
      do node = 1,isum(maxy+1)
         if (nasr_to_cell(node) .gt. 0) then
            do iasrtcell = 1,nasr_to_cell(node)
               vol_in = asr_cap_f_pws(asrnum(5,node,iasrtcell,2))
     $          * 3.069 / ncells_dest_from_asr(asrnum(5,node,
     $          iasrtcell,2))
               call asr(asrnum(5,node,iasrtcell,2),2,vol_in
     $          ,bal_vol,0,tot_cnl_dmd_asr)
               rchg(node) = rchg(node) + (vol_in-bal_vol)*43560.
     $         / gdar
               asr_t_cell_volm(node) = asr_t_cell_volm(node)
     $         + vol_in-bal_vol
            enddo
         endif
      enddo
   else
      do node = 1,isum(maxy+1)
         if (nasr_to_cell(node) .gt. 0) then
            do iasrtcell = 1,nasr_to_cell(node)
               tot_cnl_need = 0.0
               if (ncnl_dest_from_asr(asrnum(5,node,iasrtcell,2))
     $          .gt. 0) then
                  do ll = 1,ncnl_dest_from_asr(asrnum(5,node,
     $            iasrtcell,2))
                     tot_cnl_need = tot_cnl_need + dqu(cnl_dest_num
     $               (asrnum(5,node,iasrtcell,2),ll))
                  enddo
               endif
               vol_in = amin1(max(asr_outcap(asrnum(5,node,iasrtcell
     $         ,2))*3.0689-tot_cnl_need*1.9835,0.0),asr_cap_f_pws
     $         (asrnum(5,node,iasrtcell,2))* 3.0689)
     $         / ncells_dest_from_asr(asrnum(5,node,iasrtcell,2))
c              print *,month,iday,vol_in,tot_cnl_need
c    $           ,cnm(cnl_dest_num(asrnum(5,node,iasrtcell,2),1))
c    $           ,asr_outcap(asrnum(5,node,iasrtcell,2)),'  GRASR'
               call asr(asrnum(5,node,iasrtcell,2),2,vol_in
     $          ,bal_vol,0,tot_cnl_dmd_asr)
               rchg(node) = rchg(node) + (vol_in-bal_vol)*43560.
     $         / gdar
               asr_t_cell_volm(node) = asr_t_cell_volm(node)
     $         + vol_in-bal_vol
            enddo
         endif
      enddo
   endif
```

```
c
c  recovery from asr wells injected with excess water from
c   canal system and NO reservoir(e.g. C-51 ASR)
c
      DO n = nch,1,-1
         IF (nasr_to_cnl(n) .gt. 0 .and. (lcnb(n) .gt. 6.or. lcnb(n)
     $   .eq.0)) THEN
            nasrc = 0
            DO icnl = 1,nasr_to_cnl(n)
               IF (nres_src_to_asr(asrnum(1,n,icnl,2)) .eq. 0) THEN

                  nasrc = nasrc + 1
                  IF (nasrc .gt. 1) THEN
                     GO TO 30
                  ENDIF
                  vol_in = 0.0
                  DO i = 1,no_ds_cnl_ws_asr(n,icnl)
                     vol_in = vol_in + dqu(icnl_no_ws_asr(n,icnl,i))
     $               *1.9835
                  END DO
   30             call asr (asrnum(1,n,icnl,2), 2,vol_in,bal_vol,0,
     $            tot_cnl_dmd_asr)
                  IF (nint(vol_in) .gt. 0) THEN
                     ratio_del_to_need = (vol_in - bal_vol)/vol_in
                  ELSE
                     ratio_del_to_need = 0.0
                  ENDIF
                  DO j = 1,no_ds_cnl_ws_asr(n,icnl)
                     qu(icnl_no_ws_asr(n,icnl,j)) = ratio_del_to_need *
     $               dqu(icnl_no_ws_asr(n,icnl,j))*86400. + qu
     $               (icnl_no_ws_asr(n,icnl,j))
                     upstream_inflow(icnl_no_ws_asr(n,icnl,j)) =
     $               upstream_inflow(icnl_no_ws_asr(n,icnl,j)) +
     $               ratio_del_to_need*dqu(icnl_no_ws_asr(n,icnl,j))
     $               *86400.
                  END DO
                  IF (cnm(n) .eq. canl_names_spec(12)) THEN
                     vol_rec_from_c51asr = vol_rec_from_c51asr +
     $               (vol_in - bal_vol) / 1.9835
                  ELSE IF (cnm(n) .eq. canl_names_spec(22)) THEN
                     vol_rec_from_hlsbasr = vol_rec_from_hlsbasr +
     $               (vol_in - bal_vol) / 1.9835
                  ELSE IF (cnm(n) .eq. canl_names_spec(40)) THEN
                     vol_rec_from_nnrcasr = vol_rec_from_nnrcasr +
     $               (vol_in - bal_vol) / 1.9835
                  ELSE
                     vol_rec_from_l31asr = vol_rec_from_l31asr +
     $               (vol_in - bal_vol) / 1.9835
                  ENDIF
                  vol_in = bal_vol
                  DO i = 1,no_ds_cnl_ws_asr(n,icnl)
                     outflow_to_ds_cnl = ratio_del_to_need * dqu
     $               (icnl_no_ws_asr (n,icnl,i))
                     dqu(icnl_no_ws_asr(n,icnl,i)) = (1.0
     $               -ratio_del_to_need) * dqu(icnl_no_ws_asr(n,icnl,i))
                     IF (lcnb(icnl_no_ws_asr(n,icnl,i)) .gt. 6 .or. lcnb
     $               (icnl_no_ws_asr(n,icnl,i)) .eq. 0) THEN
```

```
                     asr_rec_ws(iserv_area_loc(icnl_no_ws_asr
     $               (n,icnl,i))) = asr_rec_ws(iserv_area_loc
     $               (icnl_no_ws_asr(n,icnl,i))) + outflow_to_ds_cnl
                     asr_rec_ws_cnty(icounty_loc_indx(icnl_no_ws_asr
     $               (n,icnl,i))) = asr_rec_ws_cnty(icounty_loc_indx
     $               (icnl_no_ws_asr (n,icnl,i))) + outflow_to_ds_cnl
                  ELSE
                     asr_rec_vol_wca(lcnb(icnl_no_ws_asr(n,icnl,i)))
     $               = asr_rec_vol_wca(lcnb(icnl_no_ws_asr(n,icnl,i))
     $               ) + outflow_to_ds_cnl
                  ENDIF
                  ICNL_DNS = icnl_no_ws_asr(n,icnl,i)
   40             ICNL_DNS = iup_canal_ws(ICNL_DNS)
                  IF (ICNL_DNS .LE. NCH) THEN
                     dqu(ICNL_DNS) = dqu(ICNL_DNS) -
     $               outflow_to_ds_cnl/1.9835
                     dqu(ICNL_DNS) = AMAX1(dqu(ICNL_DNS),0.0)
                     GO TO 40
                  ENDIF
               END DO
            ENDIF
         END DO
      ENDIF
   END DO
c
c
c   use asr wells in proposed Central Eastern PBC Reservoir
c         to meet western LWDD needs
c
   IF (nasr_to_cnl(ie1) .gt. 0) THEN
      DO i = 1,nasr_to_cnl(ie1)
         vol_in = (wsnde1+wsnde2+wsnde3+wsndes+wsndese) * 1.9835
         call asr (asrnum(1,ie1,i,2),2,vol_in,bal_vol, 1,
     $   tot_cnl_dmd_asr)
         IF (nint(vol_in) .gt. 0) THEN
            ratio_del_to_need = (vol_in - bal_vol) / vol_in
         ELSE
            ratio_del_to_need = 0.0
         ENDIF
         qu(ie1) = qu(ie1) + ratio_del_to_need * wsnde1 * 86400.

         qu(ie2) = qu(ie2) + ratio_del_to_need * wsnde2 * 86400.

         qu(ie3) = qu(ie3) + ratio_del_to_need * wsnde3 * 86400.

         qu(ies) = qu(ies) + ratio_del_to_need * wsndes * 86400.

         qu(iese)= qu(iese) + ratio_del_to_need * wsndese * 86400.

         wsnde1 = (1. 0- ratio_del_to_need) * wsnde1
         wsnde2 = (1. 0- ratio_del_to_need) * wsnde2
         wsnde3 = (1. 0- ratio_del_to_need) * wsnde3
         wsndes = (1. 0- ratio_del_to_need) * wsndes
         wsndese = (1. 0- ratio_del_to_need) * wsndese
         if (i .eq. 1) then
            vol_rec_from_cpbasr = vol_rec_from_cpbasr
     $      + (vol_in - bal_vol)/1.9835
```

```
            else
               vol_rec_from_acmasr = vol_rec_from_acmasr
     $         + (vol_in - bal_vol)/1.9835
            endif
            asr_rec_ws(iserv_area_loc(ie1)) = asr_rec_ws(iserv_area_loc
     $      (ie1)) + (vol_in - bal_vol)/1.9835

            asr_rec_ws_cnty(icounty_loc_indx(ie1)) = asr_rec_ws_cnty
     $      (icounty_loc_indx(ie1)) + (vol_in - bal_vol)/1.9835

         END DO
      ENDIF
c
c   use asr wells near C-51 and HILLSBORO canals to help meet
c   needs in LWDD
c
      IF (nasr_to_cnl(ic51) .gt. 0) THEN
         DO i = 1,nasr_to_cnl(ic51)
            vol_in = (wsnde1 + wsnde2 + wsnde3)*1.9835
            call asr (asrnum(1,ic51,i,2),2,vol_in,bal_vol, 1,
     $      tot_cnl_dmd_asr)
            IF (nint(vol_in) .gt. 0) THEN
               ratio_del_to_need = (vol_in - bal_vol) / vol_in
            ELSE
               ratio_del_to_need = 0.0
            ENDIF
            qu(ie1) = qu(ie1) + ratio_del_to_need * wsnde1 * 86400.

            qu(ie2) = qu(ie2) + ratio_del_to_need * wsnde2 * 86400.

            qu(ie3) = qu(ie3) + ratio_del_to_need * wsnde3 * 86400.

            wsnde1 = (1.0- ratio_del_to_need) * wsnde1
            wsnde2 = (1.0- ratio_del_to_need) * wsnde2
            wsnde3 = (1.0- ratio_del_to_need) * wsnde3
            vol_rec_from_c51asr = vol_rec_from_c51asr + (vol_in -
     $      bal_vol)/1.9835
            asr_rec_ws(iserv_area_loc(ie1)) = asr_rec_ws(iserv_area_loc
     $      (ie1)) + (vol_in - bal_vol)/1.9835

            asr_rec_ws_cnty(icounty_loc_indx(ie1)) = asr_rec_ws_cnty
     $      (icounty_loc_indx(ie1)) + (vol_in - bal_vol)/1.9835

         END DO
      ENDIF
      IF (nasr_to_cnl(lhl) .gt. 0) THEN
         DO i = 1,nasr_to_cnl(lhl)
            IF (nres_src_to_asr(asrnum(1,lhl,i,2)) .eq. 0) THEN

               vol_in = (wsndes + wsndese) * 1.9835
               call asr (asrnum(1,lhl,i,2),2,vol_in,bal_vol, 1,
     $         tot_cnl_dmd_asr)
               IF (nint(vol_in) .gt. 0) THEN
                  ratio_del_to_need = (vol_in - bal_vol) / vol_in
               ELSE
                  ratio_del_to_need = 0.0
               ENDIF
```

```
               qu(ies) = qu(ies) + ratio_del_to_need * wsndes * 86400.

               qu(iese) = qu(iese) + ratio_del_to_need * wsndese *
     $         86400.
               wsndes = (1.0- ratio_del_to_need) * wsndes
               wsndese = (1.0- ratio_del_to_need) * wsndese
               vol_rec_from_hlsbasr = vol_rec_from_hlsbasr + (vol_in-
     $         bal_vol)/1.9835
               asr_rec_ws(iserv_area_loc(lhl)) = asr_rec_ws
     $         (iserv_area_loc(lhl)) + (vol_in - bal_vol)/1.9835

               asr_rec_ws_cnty(icounty_loc_indx(lhl)) = asr_rec_ws_cnty
     $         (icounty_loc_indx(lhl)) + (vol_in - bal_vol)/1.9835

            ENDIF
         END DO
      ENDIF
c
c  asr recovery to WCAs for delivery to LEC
c
      DO n = nch,1,-1
         IF (nasr_to_cnl(n) .gt. 0 .and. (lcnb(n) .ne. 0 .and. lcnb(n)
     $   .le.6)) THEN
            nasrc = 0
            DO icnl = 1,nasr_to_cnl(n)
               IF (nres_src_to_asr(asrnum(1,n,icnl,2)) .eq. 0) THEN

                  nasrc = nasrc + 1
                  IF (nasrc .gt. 1) THEN
                     GO TO 50
                  ENDIF
                  vol_in = 0.0
                  DO i = 1,no_ds_cnl_ws_asr(n,icnl)
                     vol_in = vol_in + dqu(icnl_no_ws_asr(n,icnl,i))
     $               *1.9835
                  END DO
   50             call asr (asrnum(1,n,icnl,2), 2,vol_in,bal_vol,0,
     $            tot_cnl_dmd_asr)
                  IF (nint(vol_in) .gt. 0) THEN
                     ratio_del_to_need = (vol_in - bal_vol)/vol_in
                  ELSE
                     ratio_del_to_need = 0.0
                  ENDIF
                  DO j = 1,no_ds_cnl_ws_asr(n,icnl)
                     qu(icnl_no_ws_asr(n,icnl,j)) = ratio_del_to_need *
     $               dqu(icnl_no_ws_asr(n,icnl,j))*86400. + qu
     $               (icnl_no_ws_asr(n,icnl,j))
                     upstream_inflow(icnl_no_ws_asr(n,icnl,j)) =
     $               upstream_inflow(icnl_no_ws_asr(n,icnl,j)) +
     $               ratio_del_to_need*dqu(icnl_no_ws_asr(n,icnl,j))
     $               *86400.
                  END DO
                  vol_in = bal_vol
                  DO i = 1,no_ds_cnl_ws_asr(n,icnl)
                     outflow_to_ds_cnl = ratio_del_to_need * dqu
     $               (icnl_no_ws_asr (n,icnl,i))
                     dqu(icnl_no_ws_asr(n,icnl,i)) = (1.0
```

```
  $               -ratio_del_to_need) * dqu(icnl_no_ws_asr(n,icnl,i))
                  asr_rec_vol_wca(lcnb(icnl_no_ws_asr(n,icnl,i))) =
  $               asr_rec_vol_wca(lcnb(icnl_no_ws_asr(n,icnl,i))) +
  $               outflow_to_ds_cnl
                  ICNL_DNS = icnl_no_ws_asr(n,icnl,i)
60                ICNL_DNS = iup_canal_ws(ICNL_DNS)
                  IF (ICNL_DNS .LE. NCH) THEN
                     dqu(ICNL_DNS) = dqu(ICNL_DNS) -
  $                  outflow_to_ds_cnl/1.9835
                     dqu(ICNL_DNS) = AMAX1(dqu(ICNL_DNS),0.0)
                     GO TO 60
                  ENDIF
               END DO
            ENDIF
         END DO
      ENDIF
   END DO
   RETURN
   END
```

avail_res_stor_adjust.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: avail_res_stor_adjust.F,v 1.7 2005/06/10 13:29:00 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/avail_res_stor_adjust.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------
C   THIS SUBROUTINE COMPUTES ADJUSTMENT IN AVAILABLE STORAGE IN RESERVOIRS
C   THAT ARE NOT SEPARATE ENTITIES WITHIN GRID NETWORK. THE ADJUSTMENT IS
C   DUE TO THE DIFFERENCE BETWEEN THE ACTUAL AREA OF RESERVOIR AND AREA OF
C   THE GRID CELLS CONTAINING RESERVOIR.
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE avail_res_stor_adjust (dpthgw_for_rain, sumrain_unadj,
     & sumrain, rain, stage_res, depth_levee_spg_max_from_gw, volwtr,
     & node,ia)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'

      DIMENSION adjust_for_avail_stor(max_ncells),
     & depth_levee_spg_max_from_gw(max_ncells),
     & dpthgw_for_rain(max_ncells),
     & prev_adjust_for_avail_stor(max_ncells), rain(max_ncells),
     & stage_res(max_ncells,max_n_res_in_cell), sumrain(max_ncells),
     & sumrain_unadj(max_ncells), total_afact(max_ncells),
     & volwtr(max_n_res)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

        total_afact(node) = 0.0
        DO k = 1,noresincell(node)
```

```
          ia = ires_index_in_cell(node,k)
          IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &     (ia) .eq. 'NO ') THEN
             total_afact(node) = total_afact(node) +
     &       sfactor(ia)
          ENDIF
        ENDDO
        IF (lutyp(node).ne.0) THEN
          dpthgw_for_rain(node)=(ells(node)-h(node))*s(node)
          IF (pond(node).gt.dpthgw_for_rain(node)) THEN
            head=pond(node)-dpthgw_for_rain(node)+ells(node)
          ELSE
            head=(pond(node)-dpthgw_for_rain(node))/s(node)+ells(node)
          END IF
!
!   accumulate rainfall and ET for ecah cell for estimating stages
!   in above ground reservoirs
!
          IF (head.gt.ells(node).or.head_cell_beg(node).gt.ells(node))
     &     THEN
            sumrain_unadj(node)=sumrain_unadj(node)+rain(node)
            sumrain(node)=sumrain(node)+amax1(rain(node)-
     &       dpthgw_for_rain(node)*12.,0.0)
          END IF
!
!   compute adjustment in available storage in reservoirs due to
!    difference between actual area of reservoir and area of grid cells
!    containing reservoir
!
            DO ir = 1,noresincell(node)
             IF (ia .eq. ires_index_in_cell(node,ir)) THEN
              IF (day1) THEN
                stage_res(node,ir)=head_cell_beg(node)
              END IF
              IF (head.le.ells(node).and.head_cell_beg(node)
     &         .le.ells(node)) THEN
                afact=1.0
              ELSE
                afact=total_afact(node)
              END IF
              IF (afact.le.1.0) THEN
                sgn=1
              ELSE
                sgn=-1
              END IF
              daily_adjust_for_ovlf=daily_chg_dpth_ovl_vol(node)
              daily_adjust_for_max_vol(node)=(daily_et(node)-
     &         amax1(rain(node)/12.-dpthgw_for_rain(node),0.0)+
     &         daily_seep_loss_res(node)-chg_depth_levee_spg(node))*
     &         ((1.0-afact)/afact)-daily_outf_from_res_to_cell_max(node)
     &         /afact-daily_adjust_for_ovlf/afact
              IF (head.gt.ells(node).or.head_cell_beg(node)
     &         .gt.ells(node)) THEN
                IF (sgn*prev_adjust_for_area(node).lt.-0.000001) THEN
                  adjust_for_avail_stor(node)=
     &             daily_adjust_for_max_vol(node)+
     &             prev_adjust_for_area(node)
```

```
              adjust_for_avail_stor1=adjust_for_avail_stor(node)
            ELSE
              IF (sgn*prev_adjust_for_avail_stor(node).lt.-0.00001)
&              THEN
               adjust_for_avail_stor(node)=
&               daily_adjust_for_max_vol(node)+
&               prev_adjust_for_avail_stor(node)
              ELSE
               adjust_for_avail_stor(node)=
&               daily_adjust_for_max_vol(node)
              END IF
              adjust_for_avail_stor1=adjust_for_avail_stor(node)
            END IF
          ELSE
            adjust_for_avail_stor(node)=0.0
            adjust_for_avail_stor1=adjust_for_avail_stor(node)
          END IF
          prev_adjust_for_avail_stor(node)=adjust_for_avail_stor1
          adjust_for_avail_stor1=sgn*amax1(sgn*
&          adjust_for_avail_stor1,0.0)
          IF (head.gt.ells(node).or.head_cell_beg(node)
&          .gt.ells(node)) THEN
            add_gw_stor=depth_levee_spg_max_from_gw(node)*(1.0-
&            sfactor(ia))*gdar/86400.
          ELSE
            add_gw_stor=0.0
          END IF
          volwtr(ia)=volwtr(ia)+adjust_for_avail_stor1*sfactor(ia)*
&          gdar/86400.+add_gw_stor
          END IF
         END DO
        END IF
   volwtr(ia)=amax1(volwtr(ia),0.0)
 RETURN
 END
```

bpts.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: bpts.F,v 1.7 2003/07/03 20:07:18 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/bpts.F,v $
C   $Name: rel-5-5 $
C
C
C  inputs:  iday, month, stage, rvoir
C  outputs: mzone, stgref
C
C    - OR -
C
C  inputs:  iday, month, rvoir, mzone
C  outputs: stgref, iregjul(res, zone, brkpt)
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE bpts (iday, month, stage, rvoir, mzone, mzopt, stgref)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'lakwca.inc'

      INTEGER numd(12), rvoir

      REAL refstg(20), x(20), y(20)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  if (mod(year,4).eq.0.0) numd(2) = 29
C  note: variable "year" has to be in a common block
```

```
      xval = julian(month,iday,numd)

      if(mzopt.eq.0) then

C  determine both zone location and regulation stage
C  compute all regulation stages in all zones for this reservoir

         if (stor_area_name(rvoir) .eq. 'LOK   ') then
            max_no_fc_zones = n_fc_zones
         else
            max_no_fc_zones = nzone(rvoir)
         endif

         do 20 i = 1,max_no_fc_zones
            n=nbrkpt(rvoir,i)
            do 10 j = 1,n
               x(j) = iregjul( rvoir,i,j )
               y(j) = regstg( rvoir,i,j )
10          continue
            refstg(i) = ainterp(xval,x,y,n,0.,0.,1)
20       continue
c determine which zone stage falls into
         do 30 i = 1, max_no_fc_zones
            if (refstg(i).gt.stage) goto 30
            mzone = i
            goto 50
30       continue
         mzone = max_no_fc_zones + 1
      else
c determine regulation stage only
         n=nbrkpt(rvoir,mzone)
         do 40 j = 1,n
            x(j) = iregjul( rvoir,mzone,j )
cc    if((mod(year,4).eq.0.0).and.(month.gt.2) ) x(j) = x(j) +1
            y(j) = regstg( rvoir,mzone,j )
40       continue
         refstg(mzone) = ainterp(xval,x,y,n,0.,0.,1)
      endif
c
50    if (mzone.gt.nzone(rvoir))then
         stgref=0.0
      else
         stgref = refstg(mzone)
      endif
      return
      end
```

budg.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: budg.inc,v 1.3 2003/07/03 20:07:55 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/budg.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
      common /BUDG/ RFM(max_ncells),WELLS(max_ncells),
     &HMVOL(max_ncells,2),HMVOL_daily(max_ncells,2),
     &OVLFLO_daily(max_ncells,2)
C-----------------
```

c1.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: c1.inc,v 1.3 2003/07/03 20:07:55 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/c1.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
      common /C1/
     &rmax_stage_reuse(max_n_reuse_plnts,max_n_cells_reuse),flow_s155,
     &cout_to_c51_frm_l8bprs,rl8runoff_to_bptres,outflow_to_l8s,
     &rmin_stg_for_outf_s12(4,12),s333_ws,ftargs12(3,4,12),
     &outf_from_c11res,upstream_inflow(max_ncnls),
     &frac_struc_cap(max_ncnls,max_nostr_cnl,12),
     &LFRACBFLR(max_n_eaa_condt),opt_mrp_to_enp,opt_wca1_prop,
     &opt_l8_prop,opt_ft_input,opt_st_input,BFLOW_S352,
     &frac_bkflw_wpb_via_l8_bel_flr,bflo_via_l8

      LOGICAL LFRACBFLR,opt_mrp_to_enp,opt_wca1_prop,opt_l8_prop,
     $opt_ft_input,opt_st_input
      DOUBLE PRECISION rl8runoff_to_bptres
```

c2.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     SCCS ID: %W% %G% SFWMD Planning Department
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: c2.inc,v 1.2 2003/07/03 20:07:56 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/c2.inc,v $
C       $Name: rel-5-5 $
C
C ----------------------------------------------------

      common /C2/
     &earth_plug_opt,resname_reuse(max_n_reuse_plnts,max_n_res_reuse),
     &LVNAME(max_no_lvsp),FLNM(max_n_struc_flw),CNM(max_ncnls),
     &PLTNM(max_n_gage_loc_outp,max_n_cell_loc_gage),B,DATADIR,TYPERUN,
     &EAACONV,LCALCPT(max_n_str_node_dep),land_use_type(max_n_lu_type),
     &type_flow_across_ttrail,simcaes,simsles,type_target,wca3aenv,
     &wca2aenv,wca1env,wcaid(max_n_wcas),wca_import_variation,
     &nnrctwca3a,nesrsenv,nwparkenv,eaa_conv_opt_s8,type_s355,
     &eaa_conv_opt_s7,hlyenv,rotenenv,bcnpenv,ws_need_l31nc_orig,l31nc,
     &loxenv,ssm_env,flow_to_wca_prior,regso_opt,s10env,s11env,
     &wca_import_min,res_out_strname(max_n_res,max_nostr_res),
     &borient(max_ncells,5),res_out_type(max_n_res,max_nostr_res),
     &floor_grid_cell_opt,holy_oper,ws_to_holy_opt,
     &gage_name(max_n_wcas,9,3),demand_level_opt,divers_excess_to_res,
     &opt_for_hill_bypass,opt_for_hill_bypass_runoff,holy_outflow_opt,
     &ws_status_cnl(max_ncnls),runoff_to_holeyland,holey_out_flow_pump,
     &holey_min_level_opt,opt_reg_releases_to_lec,runmode,
     &opt_flow_to_ts,make_up_water_opt,main_pres_level_fl_prot_ts,
     &use_enp_ws_to_lec,s343sparrow_opt,es_dmnd_acc_freq,
     &lok_reg_to_wca_mode,wca_name(max_n_wcas),s344sparrow_opt,
     &back_sp_pump_s9_opt,makeup_water_restr,s332sparrow_opt,wca3zone,
     &opt_ws_to_roten_frm_lok,opt_outflow_from_res_to_wca,
     &opt_multi_seas_for_reg_wca,opt_fc_canal(max_ncnls),
     &sta34_outf_flex_eaar_opt,
     &opt_for_reg_rel_to_res(max_n_eaa_condt,max_n_reg_rel_basin)

      CHARACTER res_out_strname*6,borient*1,res_out_type*6,loxenv*5
     $ ,earth_plug_opt*3,resname_reuse*6,LVNAME*5,FLNM*6,CNM*5,PLTNM*6
```

```
$ ,B*300,DATADIR*200,TYPERUN*7,EAACONV*7,LCALCPT*5,land_use_type*7
$ ,type_flow_across_ttrail*7,simcaes*3,simsles*3,type_target*5
$ ,wca2aenv*5,wca1env*5,wcaid*5,wca_import_variation*5
$ ,nnrctwca3a*5,nesrsenv*5,nwparkenv*5,eaa_conv_opt_s8*4
$ ,wca3aenv*5,type_s355*4,eaa_conv_opt_s7*4,hlyenv*5,rotenenv*5
$ ,bcnpenv*5,ssm_env*5,flow_to_wca_prior*3,regso_opt*7,s10env*5
$ ,s11env*5,wca_import_min*5,floor_grid_cell_opt*5,holy_oper*5
$ ,ws_to_holy_opt*6,gage_name*6,demand_level_opt*4
$ ,divers_excess_to_res*5,opt_for_hill_bypass*5,wca_name*5
$ ,opt_for_hill_bypass_runoff*5,holy_outflow_opt*4,ws_status_cnl*5
$ ,runoff_to_holeyland*7,holey_out_flow_pump*5
$ ,holey_min_level_opt*3,opt_reg_releases_to_lec*5,runmode*5
$ ,opt_flow_to_ts*7,main_pres_level_fl_prot_ts*5
$ ,use_enp_ws_to_lec*5,es_dmnd_acc_freq*3,lok_reg_to_wca_mode*4
$ ,make_up_water_opt*7,s343sparrow_opt*5,back_sp_pump_s9_opt*3
$ ,opt_for_reg_rel_to_res*5,s344sparrow_opt*5
$ ,opt_ws_to_roten_frm_lok*3,makeup_water_restr*5,s332sparrow_opt*5
$ ,wca3zone*1,opt_outflow_from_res_to_wca*4,opt_fc_canal*6
$ ,opt_multi_seas_for_reg_wca*3,sta34_outf_flex_eaar_opt*3
```

caloos.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: caloos.F,v 1.13 2005/06/10 13:29:00 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/caloos.F,v $
C   $Name: rel-5-5 $
C
C------------------------------------------------------------------------------
C  This subroutine determines the interaction (inflows/outflows)
C  between Lake Okeechobee,Caloosahatchee basin,and Caloosahatchee
C  estuary.
C------------------------------------------------------------------------------
C  definition of terms: (all units in ac-ft/day)
C  input:
C       c43dmd      = c43 basin demand (adjusted for s235 inflow)
C       c43ro       = c43 basin runoff (adjusted for s235 inflow)
C        note: s235 = 0.0,   IF stage_lok < 13.0 ngvd
C                   = 0.84 * s4 basin runoff,   otherwise
C       caedmd      = caloosahatchee estuarine demand
C       cbcaest     = fraction of C43 estuarine demand to be met
C                     (based on ssm, for now)
C       cbc43       = fraction of c43 basin demand to be met
C                     as calculated in the supply-side management
C                     subroutine
C       stglok      = lake okeechobee stage
C
C  output:
C       bsnsto      = portion of c43 basin runoff that has to stay as
C                     storage in the basin since outlet capacities
C                     are already exceeded
C       unmetdmd    = portion of c43 basin demand that cannot be met
C                     by LOK
C
C       ealc43ro    = c43 basin runoff used to meet estuarine demand
C       eallok1     = discharge from LOK used to meet estuarine demand
C       rmin_lok_stg_del_est = minimum LOK stage allowed to meet
C                              estuarine demand
```

```
C       alc43       = c43 basin allocation;
C                     c43 basin demand multiplied by the cutback cbc43
C       eflc43      = c43 basin runoff in excess of estruarine demand
C                     which can go to LOK or the estuarine
C       eallok2     = discharge from LOK used to meet c43 basin demand
C       eqs79       = total flow thorugh s79 (excludes regulatory flows)
C       eqs77       = total flow thorugh s77 (excludes regulatory flows)
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE caloos1(iyr, imo, ida, c43dmd, c43ro, caedmd, cbcaest,
     & cbc43, stglok, bsnsto, unmetdmd, dmd_met_by_lok, eallok1, stgref,
     & rmulti_seas_pred, clim_threshold_est, n_fc_zones,caldmndro,jday,
     & dmnd_orig,ro_orig)
C-----------------------------------------------------------------------------
C  This subroutine determines the interaction (inflows/outflows)
C  between Lake Okeechobee,Caloosahatchee basin,and Caloosahatchee
C  estuary on a daily basis. This requires daily time series of basin
C  demand/runoff and a daily time series of estuarine demands.
C-----------------------------------------------------------------------------

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'ccaloos.inc'
      INCLUDE 'opersched.inc'

      DIMENSION clim_threshold_est(2), rmulti_seas_pred(12),
     &caldmndro(2,366)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C initialize allocations and estimated deliveries

      alcae = 0.0
      ealc43ro = 0.0
      eallok1 = 0.0
      c43est_met_by_lok = 0.0
      alc43 = 0.0
      eflc43 = 0.0
      eallok2 = 0.0
      c43dmd_met_by_lokres = 0.0
      dmd_met_by_lok = 0.0
      unmetdmd = 0.0
      excess_ro_to_est = 0.0
      excess_ro_to_lok = 0.0
      excess_ro_to_sto = 0.0
      c43ro0 = c43ro

      CALL bpts(IDA,IMO,stglok,1,mzone,0,stgref_est)
      CALL bpts(ida,imo,10.0,1,n_fc_zones+6, 1,
     $rmin_lok_stg_del_est_normf)

c calculate caloosahatchee estuary allocation

      alcae = cbcaest*caedmd
```

```
      alcae0 = alcae
      IF (alcae .eq. 0.0) goto 10
c estimate portion of allocation to be met by c43 basin runoff
      ealc43ro = amin1( alcae, c43ro)
      IF (ealc43ro .eq. alcae) THEN
         c43ro = c43ro - ealc43ro
         alcae = 0.0
      ELSE
         alcae = alcae - c43ro
         c43ro = 0.0
      ENDIF
c update estimated structure flow
      eqs79 = eqs79 + ealc43ro
      s79cap = s79cap - ealc43ro
c estimate portion of allocation to be met by Lake Okeechobee
c assumption: LOK is always capable to meet remaining estuarine demand
      IF ((stglok .gt. rmin_lok_stg_del_est .and. iclimate_opt .eq. 0)
     $.or. (stglok .gt. rmin_lok_stg_del_est .and. ((rmulti_seas_pred
     $(imo) .gt. clim_threshold_est(1) .and. stglok .ge.
     $rmin_lok_stg_del_est_normf .and. mzone .ge. mzone_puls_top).or.
     $(rmulti_seas_pred(imo) .gt. clim_threshold_est(2) .and. stglok
     $.lt. rmin_lok_stg_del_est_normf) .or. (mzone .lt. mzone_puls_top
     $.and. no_puls_zones.gt.0) .or. (mzone .le. n_fc_zones .and.
     $no_puls_zones.eq.0)))) THEN


         eallok1 = alcae
         eallokres1 = 0.0
      ELSE
         eallok1 = 0.0
         IF (lok_split_option .eq.  1.and. stagelo_res .ge. 2.0) THEN
            eallokres1 = alcae
         ELSE
            eallokres1 = 0.0
         ENDIF
      ENDIF
c update estimated structure flows
      eqs77 = eqs77 + eallok1 + eallokres1
      s77cap = s77cap - eallok1 - eallokres1
      eqs79 = eqs79 + eallok1 + eallokres1
      s79cap = s79cap - eallok1 - eallokres1
      c43est_met_by_lok = eallok1 + eallokres1
c
c excess C-43 BASIN runoff accounting
c
c determine where excess runoff from c43 basin will go after
c estuary's allocations have been met
   10 IF (c43ro .ne. 0.0) THEN
         eflc43 = c43ro
         CALL bpts(ida,imo,10.0,1,n_fc_zones + 9,1,stg_backflw_cal)
         IF (stglok .gt. stg_backflw_cal) THEN
c LOK stage too high, release to tide
            excess_ro_to_est = eflc43
            eqs79 = eqs79 + eflc43
            s79cap = s79cap - eflc43
            IF (s79cap .lt. 0.0) THEN
c capacity of s79 exceeded, portion of excess runoff stays in C-43 basin
```

```
c put back flows
               s79cap = s79cap + eflc43
               excess_ro_to_est = s79cap
               eqs79 = eqs79 - eflc43
c new s79 flow
               eqs79 = s79cap0
c or eqs79 = eqs79 + s79cap
c basin storage is excess runoff minus remaining capacity of s79
               bsnsto = eflc43 - s79cap
               excess_ro_to_sto = bsnsto
c remaining capacity of s79 is zero
               s79cap = 0.0
            ENDIF
         ELSE
c LOK stage is low enough, backflow to LOK possible
c might be limited by operational constraints, e.g.
c     may not be possible when lake stage is higher
c     than tailwater of S77
c assumption: maximum backflow is equal to design Q
            excess_ro_to_lok = eflc43
            eqs77 = eqs77 - eflc43
            s77cap = s77cap - eflc43
            eeflc43 = 0.0
            IF (s77cap .lt. 0.) THEN
c capacity of s77 exceeded, portion of excess runoff goes to tide
c put back flows
               s77cap = s77cap + eflc43
               excess_ro_to_lok = s77cap
               eqs77 = eqs77 + eflc43
c new s77 flow
               eqs77 = -1.0*s77cap0
c remainder of excess runoff can go to tide
               eeflc43 = eflc43 - s77cap
               excess_ro_to_est = eeflc43
c remaining capacity of s77 is zero
               s77cap = 0.0
            ENDIF
c release rest of flow to tide, IF permissible
            eqs79 = eqs79 + eeflc43
            s79cap = s79cap - eeflc43
            IF (s79cap .lt. 0.0) THEN
               s79cap = s79cap + eeflc43
               excess_ro_to_est = s79cap
               eqs79 = eqs79 - eeflc43
               eqs79 = s79cap0
c or eqs79 = eqs79 + s79cap
               bsnsto = eeflc43 - s79cap
               excess_ro_to_sto = bsnsto
               s79cap = 0.0
            ENDIF
         ENDIF
      ENDIF
c
c C-43 BASIN demand accounting
c
c calculate c43 basin allocation
c
```

```
      alc43 = cbc43*c43dmd
      IF (alc43 .eq. 0.0) goto 20
c entire allocation is to be met by Lake Okeechobee, IF permissible
      eallok2 = alc43
c update estimated structure flow
      dmd_met_by_lok = eallok2
      eqs77 = eqs77 + eallok2
      IF (lok_split_option .eq. 1) THEN
         IF (stglok .le. stgref .and. stagelo_res .ge. 2.0) THEN
            c43dmd_met_by_lokres = eallok2
         ELSE
            c43dmd_met_by_lokres = 0.0
         ENDIF
      ELSE
         c43dmd_met_by_lokres = 0.0
      ENDIF
      s77cap = s77cap - eallok2
      IF (s77cap .lt. 0.0) THEN
         s77cap = s77cap + eallok2
         dmd_met_by_lok = s77cap
         eqs77 = eqs77 - eallok2
c
         eqs77 = s77cap0
c or eqs77 = eqs77 + s77cap
         unmetdmd = eallok2 - s77cap
         c43dmd_met_by_lokres = c43dmd_met_by_lokres - unmetdmd
         c43dmd_met_by_lokres = amax1(c43dmd_met_by_lokres,0.0)
         s77cap = 0.0
      ENDIF
c
c WRITE summary for today
   20 WRITE(73,30)iyr,imo,ida,caldmndro(2,jday),dmnd_orig
     $,c43dmd,alc43,dmd_met_by_lok,unmetdmd,caldmndro(1,jday),ro_orig,
     $c43ro0,excess_ro_to_est,excess_ro_to_lok, excess_ro_to_sto,alcae0,
     $ealc43ro,eallok1,eqs77,eqs79
clb      ENDIF
      RETURN
   30 FORMAT(3i5,24f12.3)
      END
      subroutine caloos2(iyr,imo,ida,ndays,c43dmd,c43ro,caedmd, cbc43,
     $stglok,lok_supp_day,bflo_cap,bflo_frac, lok_supp_c43est,
     $c43dmd_met_by_lok, iuse1_asr_first,bsn2est_af, stgref,
     $rmulti_seas_pred, clim_threshold_est,n_fc_zones,cbcaest,
     $caldmndro,jday,dmnd_orig,ro_orig)
c
c-------------------------------------------------------------
c  This subroutine determines the interaction (inflows/outflows)
c  between Lake Okeechobee,Caloosahatchee basin,and Caloosahatchee
c  estuary on a monthly basis. This requires daily time series of basin
c  demand/runoff and a monthly time series of estuarine demand.
c
c-------------------------------------------------------------
      include 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'ccaloos.inc'
c
      include 'opersched.inc'
```

```
c
      DIMENSION rmulti_seas_pred(12),clim_threshold_est(2),
     &caldmndro(2,366)

c-------------------------------------------------------------
c
c definition of terms: (all units in ac-ft/day)
c input:
c       c43dmd      = c43 basin demand (adjusted for s235 inflow)
c       c43ro       = c43 basin runoff (adjusted for s235 inflow)
c        note: s235 = 0.0,   IF stage_lok < 13.0 ngvd
c                   = 0.84 * s4 basin runoff,   otherwise
c       caedmd      = caloosahatchee estuarine demand
c       cbc43       = fraction of c43 basin demand to be met
c                     as calculated in the supply-side management
c                     subroutine
c       stglok      = lake okeechobee stage
c
c output:
c       eallok1     = discharge from LOK used to meet estuarine demand
c       rmin_lok_stg_del_est = minimum LOK stage allowed to meet
c                              estuarine demand
c       alc43       = c43 basin allocation;
c                     c43 basin demand multiplied by the cutback cbc43
c       eflc43      = c43 basin runoff in excess of estruarine demand
c                     which can go to LOK or the estuarine
c       eqs79       = total flow thorugh s79 (excludes regulatory flows)
c       eqs77       = total flow thorugh s77 (excludes regulatory flows)
c
      REAL lok_supp0_day,lok_supp_day,bflo_cap,bflo_frac
      REAL lok_supp_c43est
c initialize allocations and estimated deliveries
      eallok1           = 0.0
      c43est_met_by_lok = 0.0
      alc43             = 0.0
      eflc43            = 0.0
c      c43dmd_met_by_lok = 0.0
      ro_to_lok         = 0.0
      ro_to_estuary     = 0.0
c     eqs79             = 0.0
c     eqs77             = 0.0
      caedmd0           = caedmd
      c43ro0            = c43ro
      IF (ida.eq.1) THEN
         lok_supp0_day = lok_supp_c43est/ndays
      ENDIF
c
      CALL bpts(IDA,IMO,stglok,1,mzone,0,stgref_est)
      CALL bpts(ida,imo,10.0,1,n_fc_zones+6, 1,
     $      rmin_lok_stg_del_est_normf)
c
c note: caedmd is the estuarine demand to be met by Lake Okeechobee
c       and c43 basin.  It can be cutback by using the cutback fraction
c       from ssm or by other means but will not be subject to any cutback
c       in this version because the target supplemental deliveries are
c       preprocessed based on known runoff/demand conditions in Caloosahatchee
c       basin
```

```
c
      IF (lok_supp_day.gt.0.) THEN
c supplementary release from LOK required
         IF ((stglok .gt. rmin_lok_stg_del_est .and. iclimate_opt .eq.
     $   0) .or. (stglok .gt. rmin_lok_stg_del_est .and.
     $   ((rmulti_seas_pred(imo) .gt. clim_threshold_est(1) .and.
     $   stglok .ge. rmin_lok_stg_del_est_normf .and. mzone .ge.
     $   mzone_puls_top).or. (rmulti_seas_pred(imo) .gt.
     $   clim_threshold_est(2) .and. stglok .lt.
     $   rmin_lok_stg_del_est_normf) .or. (mzone .lt. mzone_puls_top
     $   .and. no_puls_zones.gt.0) .or. (mzone .le. n_fc_zones .and.
     $   no_puls_zones.eq.0)))) THEN

            eallok1 = lok_supp_day*cbcaest
            eallokres1 = 0.0
         ELSE
            eallok1 = 0.0
            IF (lok_split_option .eq. 1.and. stagelo_res .ge. 2.0) THEN
               eallokres1 = lok_supp_day
            ELSE
               eallokres1 = 0.0
            ENDIF
         ENDIF
c note: eallok1 can be made subject to cutback ==> do this in future versions
c update estimated structure flows
         eqs79 = eqs79 + eallok1 + eallokres1
         s79cap = s79cap - eallok1 - eallokres1
         eqs77 = eqs77 + eallok1 + eallokres1
         s77cap = s77cap - eallok1 - eallokres1
         c43est_met_by_lok = eallok1 + eallokres1
      ELSE
         eallokres1 = 0.0
      ENDIF
c
c C-43 basin runoff, IF any, can backflow, IF LOK stage is low enough, or go to
c estuary as excess basin runoff (note: This runoff may be considered as excess
c or partial excess relative to today's estuarine demand but may all be needed
c to satisfy the estuarine's total monthly demand.)
      IF (c43ro .gt. 0.) THEN
         eflc43 = c43ro
         CALL bpts(ida,imo,10.0,1,n_fc_zones + 9,1,stg_backflw_cal)
         IF (stglok .gt. stg_backflw_cal) THEN
c LOK stage too high, release to tide
            ro_to_estuary = eflc43
            eqs79 = eqs79 + eflc43
            s79cap = s79cap - eflc43
         ELSE
c LOK stage is low enough, backflow to LOK possible
            IF (bflo_cap .ge. bflo_frac*eflc43) THEN
c release all runoff to LOK within allowable fraction
               ro_to_lok = bflo_frac*eflc43
               eqs77 = eqs77 - ro_to_lok
               s77cap = s77cap - ro_to_lok
c               s77cap = s77cap + ro_to_lok
               bflo_cap = bflo_cap - ro_to_lok
c release remaining runoff to estuary, assume no c43 basin runoff stays in stora
               ro_to_estuary = eflc43 - ro_to_lok
```

```
               eqs79 = eqs79 + ro_to_estuary
               s79cap = s79cap - ro_to_estuary
            ELSE
c release portion of runoff to LOK up to bflo_cap within allowable fraction
               ro_to_lok = amin1(bflo_frac*eflc43,bflo_cap)
               eqs77 = eqs77 - ro_to_lok
               s77cap = s77cap - ro_to_lok
c               s77cap = s77cap + ro_to_lok
               IF (ro_to_lok .eq. bflo_cap) THEN
c backflow capacity is reduced to zero
                  bflo_cap = 0.0
               ELSE
                  bflo_cap = bflo_cap - ro_to_lok
               ENDIF
c release remaining runoff to estuary
               ro_to_estuary = eflc43 - ro_to_lok
               eqs79 = eqs79 + ro_to_estuary
               s79cap = s79cap + ro_to_estuary
            ENDIF
         ENDIF
      ENDIF
c
c     IF (iuse1_asr_first.eq.1.and.c43dmd.gt.0.) THEN
      alc43 = cbc43*c43dmd
c entire allocation to be met by Lake Okeechobee
      c43dmd_met_by_lok = alc43
c update structure flow
      IF (lok_split_option .eq. 1) THEN
         IF (stglok .le. stgref .and. stagelo_res .ge. 2.0) THEN
            c43dmd_met_by_lokres = alc43
         ELSE
            c43dmd_met_by_lokres = 0.0
         ENDIF
      ELSE
         c43dmd_met_by_lokres = 0.0
      ENDIF
      eqs77 = eqs77 + c43dmd_met_by_lok
      s77cap = s77cap - c43dmd_met_by_lok
c     ENDIF
c
c WRITE summary for today
   10 WRITE(73,20)iyr,imo,ida,caldmndro(2,jday),dmnd_orig
     $,c43dmd,alc43,c43dmd_met_by_lok,caldmndro(1,jday),ro_orig,c43ro0
     $,ro_to_lok,ro_to_estuary,stglok,caedmd0, lok_supp0_day
     $,lok_supp_day,bflo_cap,eqs77,eqs79, bsn2est_af

clb      ENDIF
      RETURN
   20 FORMAT(3i5,24f12.3)
      END
```

canalloc.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: canalloc.inc,v 1.4 2003/07/03 20:07:56 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/canalloc.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      common /canalloc/ xcn(max_ncnls,max_ncnlloc),
     &ycn(max_ncnls,max_ncnlloc),icl(max_ncnls,max_ncnlloc)
C---------------------
      INTEGER xcn,ycn
```

canl_dep_struc_capac_setup.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: canl_dep_struc_capac_setup.F,v 1.19 2005/06/10 13:29:01 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/canl_dep_struc_capac_setup.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE DETERMINES THE AVAILABLE CAPACITY OF OUTLET STRUCTURES
C   FROM CANAL, TAKING INTO ACCOUNT OPEN/CLOSE CRITERIA, AVAILABLE VOLUME
C   OF WATER UPSTREAM OF STRUCTURES,AND TAILWATER CONSTRAINTS.
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE canl_dep_struc_capac_setup (n, ind1, idncl, conv,
     & crmin, iter, rain, prevqu, volwtr, tempf_volwtr, jday, tdqu_dn,
     & tot_flw_to_c9r_frm_nlkblt, rnlkblt_to_s29, rlvseep_to_cnl,
     & is334_open_flg, s334_fc, capacfl, capacws, tempf_weircff, wsdmnd,
     & imgt, hws, tws, stgdiff_canal_tw, gravity_flow_struc,
     & flow_to_c9r_frm_c11, tot_grav_cap,canl_rf_avg,s337fc)

      INCLUDE  'wmm_par.inc'
      INCLUDE  'abc.inc'
      INCLUDE  'c1.inc'
      INCLUDE  'c2.inc'
      INCLUDE  'routc.inc'
      INCLUDE  'stat.inc'
      INCLUDE  'wrout.inc'
      INCLUDE  'wcaot.inc'
      INCLUDE  'stas.inc'
      INCLUDE  'agdata.inc'
      INCLUDE  'lakwca.inc'
      INCLUDE  'et.inc'
      INCLUDE  'environ.inc'
      INCLUDE  'resadj.inc'
      INCLUDE  'wcasim.inc'
      INCLUDE  'resws.inc'
```

```
      INCLUDE  'opersched.inc'
      INCLUDE  'lokres.inc'
      INCLUDE  'resbudg.inc'
      INCLUDE  'stormd.inc'
      INCLUDE  'dualops.inc'

      DIMENSION capacfl(max_n_seas,max_n_substr), capacws(max_n_seas),
     & crmin(max_ncnls,max_n_seas), prevqu(max_ncnls), rain(max_ncells),
     & rlvseep_to_cnl(max_ncnls), tdqu_dn(max_ncnls),
     & tempf_volwtr(max_n_res), tempf_weircff(max_n_res,max_nostr_res),
     & volwtr(max_n_res),canl_rf_avg(max_ncnls),stg_trig_on_off(50)

      DOUBLE PRECISION crmin,hw,tw,hws,tws,rminstg_f,
     &rdn_stg_lim,rdn_stg_lim_max,rdn_stg_lim_ws,
     &diff_stg_frm_dnstg,diff_stg_frm_dnstg_ws,rminstg
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      hw=hws
      tw=tws
      IF (flnm(istr(n,ind1)).eq.strname_spec(67)) THEN
        tot_grav_cap=0.0
      END IF
      iflo=istr(n,ind1)
      stg_trig_fc_ops(n,ind1)=hw
      idiff_trigc_upstrmc = 0
      DO isfc_ops = 1,nsfc_w_chg_trg
         IF (n.eq.icnl_no_upstrm_trg_ops(isfc_ops).and.ind1.eq.
     &   istruc_no_chg_trg_ops(isfc_ops)) THEN
            IF (istruc_no_alt_trg_ops(isfc_ops) .gt. 0) THEN
               IF (nint(kflo(istruc_no_alt_trg_ops(isfc_ops))).eq. 0)
     &         THEN
                  itrg_indx = 2
               ELSE
                  itrg_indx = 1
               ENDIF
            ELSE
              itrg_indx = 1
            ENDIF
            IF (icanl_no_trg_ops(isfc_ops,itrg_indx) .le. nch) then
              IF (node_pos_trg_ops(isfc_ops,itrg_indx) .gt. 0) THEN
                IF (nodcr(icanl_no_trg_ops(isfc_ops,itrg_indx)).gt.1)
     &          THEN
                 canl_stg_diff_frm_dnstrm = float(nodcr
     &           (icanl_no_trg_ops(isfc_ops,itrg_indx))-node_pos_trg_ops
     &           (isfc_ops,itrg_indx))/float(nodcr(icanl_no_trg_ops
     &           (isfc_ops,itrg_indx))-1)*hdc(icanl_no_trg_ops(isfc_ops,
     &           itrg_indx),iseas)
                ELSE
                 canl_stg_diff_frm_dnstrm = 0.0
                ENDIF
              ELSE
                canl_stg_diff_frm_dnstrm = 0.0
              ENDIF
               stg_trig_fc_ops(n,ind1) = chdep(icanl_no_trg_ops
     &         (isfc_ops,itrg_indx)) + canl_stg_diff_frm_dnstrm
               IF (icnl_no_upstrm_trg_ops(isfc_ops) .ne.
     &         icanl_no_trg_ops(isfc_ops,itrg_indx))
```

```
     &         idiff_trigc_upstrmc = 1
            ELSE
               stg_trig_fc_ops(n,ind1) =h(igrid_loc_trg_ops(isfc_ops
     &         ,itrg_indx))+pond(igrid_loc_trg_ops(isfc_ops,itrg_indx))
               idiff_trigc_upstrmc = 1
            ENDIF
c           if (conv.eq. 1 .and. flnm(istr(n,ind1)).eq.'S9A ') print *,
c    &      stg_trig_fc_ops(n,ind1),itrg_indx,
c    &      istruc_no_alt_trg_ops(isfc_ops),cnm(n),flnm(iflo),' LKLKLK '
c    &      ,icanl_no_trg_ops(isfc_ops,itrg_indx)
c    &      ,flnm(istruc_no_alt_trg_ops(isfc_ops))
c    &      ,chdep(icanl_no_trg_ops(isfc_ops,itrg_indx))
c    &      ,canl_stg_diff_frm_dnstrm
         ENDIF
      ENDDO
      DO 20 ns=1,nsubstr(n,ind1)
C
        IF (irating(n,ind1,ns,imgt).eq.1) THEN
          deltah=hw-tw
          IF (deltah.gt.rminhd(n,ind1)) THEN
            capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,month)*
     &       (deltah-rminhd(n,ind1))**pwr(n,ind1,ns,imgt)
            IF (ides_cap_limit(n,ind1).eq.1) THEN
              IF (flnm(iflo).ne.'S334FC') THEN
                capac=amin1(capac,strdcap(n,ind1,ns,imgt))
              ELSE
                dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
                IF (pond(node_g3273).gt.dpthgw) THEN
                  stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
                ELSE
                  stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
     &             ells(node_g3273)
                END IF
                DO ncal = 1,nch
                  DO i=1,nstrctr(ncal)
                    IF (tmpname(ncal,i).eq.'S333  ') THEN
                      istruc_s333=i
                      incl_s333 = ncal
                    ENDIF
                  END DO
                  ENDDO
                DO i=1,nstrctr(ica3)
                  IF (tmpname(ica3,i).eq.'S333  ') istruc_s333=i
                END DO
                IF (stage_g3273.le.stg_trigger_fl(incl_s333,istruc_s333
     &           ,1).or.is334_open_flg.ne.0.or.frac_struc_cap(n,ind1
     &           ,month).eq.1.0) THEN
                  capac=amin1(capac,strdcap(n,ind1,ns,imgt))
                END IF
              END IF
            END IF
          ELSE
            capac=0.0
          END IF
          lpump(n,ind1,ns)=.false.
          if (flnm(iflo).eq.strname_spec(37)) THEN
               lgrav(n,ind1,ns)=.TRUE.
```

```
                 rmxstg(n,ind1,iseas) = -901.
            endif
c           if (cnm(n).eq.'C4 '.and.conv.eq.1.) print *,lgrav(n,ind1,ns)
c     &     ,flnm(iflo),'  KKKK'
            if (flnm(iflo).eq.'S335 ') lgrav(n,ind1,ns)=.TRUE.
            IF (stg_trig_fc_ops(n,ind1).gt.rmxstg(n,ind1,iseas).or.
     &      lgrav(n,ind1,ns)) THEN
              IF (.not.lgrav(n,ind1,ns)) THEN
                IF (stg_trig_fc_ops(n,ind1).gt.rmxstg(n,ind1,iseas)
     &          + dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                  capacfldc=capac
                ELSE
                  fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                  capacfldc=fmultipl*capac*
     &            (stg_trig_fc_ops(n,ind1)-rmxstg(n,ind1,iseas))
                END IF
              ELSE
                capacfldc=capac
              END IF
              IF (conv.eq.1) lgrav(n,ind1,ns)=.true.
              IF (conv.eq.1 .and. flnm(istr(n,ind1)).ne.strname_spec(37))
     &        THEN
                if (stg_trig_fc_ops(n,ind1) .ge. rmnstgs(n,ind1,iseas))
     &          icount_bel_fc_thres(n,ind1,ns) = 0
                lgrav(n,ind1,ns)=.not.(stg_trig_fc_ops(n,ind1).lt.
     &           rmnstgs(n,ind1,iseas).and.canl_rf_avg(n) .lt. 3.80/14.
     &           .and. icount_bel_fc_thres(n,ind1,ns) .ge. 5)
                if (stg_trig_fc_ops(n,ind1) .lt. rmnstgs(n,ind1,iseas)
     &           .and.canl_rf_avg(n) .lt. 3.80/14.) icount_bel_fc_thres
     &           (n,ind1,ns) = icount_bel_fc_thres(n,ind1,ns) + 1
              END IF
            ELSE
              capacfldc=0.0
              if (conv.eq.1.) icount_bel_fc_thres(n,ind1,ns) =
     &        icount_bel_fc_thres(n,ind1,ns) + 1
            END IF
c           if (flnm(iflo).eq.'S177 ' .and. conv.eq.1.) print *,
c     &     stg_trig_fc_ops(n,ind1),rmxstg(n,ind1,iseas),
c     &     rmnstgs(n,ind1,iseas),icount_bel_fc_thres(n,ind1,ns),' S177C '
c     &     ,lgrav(n,ind1,ns)
            if (flnm(iflo).eq.'S335 ' .and. nint(s337fc) .gt. 0) capacfldc
     &       = min(capacfldc,s337fc)
c           if (flnm(iflo).eq.'S381  '.and.conv.eq.1.) print *,capacfldc
c     &     ,stg_trig_fc_ops(n,ind1),hw,tw,'  S381HWTW  ',lgrav(n,ind1,ns)
c           if (conv.eq.1.) then
c           if (flnm(istr(n,ind1)).eq.'S155  '.or. flnm(istr(n,ind1))
c     &      .eq. 'S40  '.or. flnm(istr(n,ind1)).eq.'S41  ') write(445,*)
c     &      hw,tw,rmxstg(n,ind1,iseas),rmnstgs(n,ind1,iseas),
c     &      capacfldc,chdep(n),'   C51O  ',lgrav(n,ind1,ns),
c     &      stg_trig_fc_ops(n,ind1),capac,rminhd(n,ind1),month,iday
c           endif
            IF (flnm(istr(n,ind1)).eq.strname_spec(67)) tot_grav_cap=
     &       capac+tot_grav_cap
C
          ELSE IF (irating(n,ind1,ns,imgt).eq.2) THEN
            IF (ingvhdopt(n,ind1,ns).eq.1) THEN
              IF (hw.le.tw) THEN
```

```
            capacp=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &       month)
          ELSE
            capacp=0.0
            IF (conv.eq.1) lpump(n,ind1,ns)=.false.
          END IF
        ELSE
          capacp=amax1(discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &     month)-gravity_flow_struc,0.0)
        END IF
        IF (idiff_trigc_upstrmc .eq. 0) THEN
           trg_stg_hw = stg_trig_fc_ops(n,ind1)
           diff_stg_frm_dnstg = trg_stg_hw - chdep(n)
        ELSE
           trg_stg_hw = hw
           diff_stg_frm_dnstg = hw - chdep(n)
        ENDIF
        rdn_stg_lim_max = rmnstgs(n,ind1,iseas)-diff_stg_frm_dnstg
        diff_stg_frm_dnstg_ws = hw - chdep(n)
        rdn_stg_lim_ws = stgintke(n,ind1,ns,imgt)
     &   - diff_stg_frm_dnstg_ws
c
        lgrav(n,ind1,ns)=.false.
        DO iflwth=1,no_flwth_cnls
          IF (n.eq.icnl_flwth(iflwth)) THEN
            IF (iup_canal_ws(idncl).ne.n) THEN
              DO istruc=1,nstrctr(n)
                IF (nint(tdqu_dn(idncnl_ws(n,istruc,1))).gt.0) THEN
                  capacp=0.0
                END IF
              END DO
              GO TO 10
            END IF
          END IF
        END DO
   10   IF (stg_trig_fc_ops(n,ind1).gt.rmxstg(n,ind1,iseas).or.
     &  lpump(n,ind1,ns)) THEN
          if (conv.eq.1. .and. stg_trig_fc_ops(n,ind1) .ge.
     &    rmnstgs(n,ind1,iseas)) icount_bel_fc_thres(n,ind1,ns) = 0
          IF (.not.lpump(n,ind1,ns)) THEN
            IF (idiff_trigc_upstrmc .ne. 0) then
               stg_trig_f_vol = max(stg_trig_fc_ops(n,ind1),hw)
            ELSE
               stg_trig_f_vol = stg_trig_fc_ops(n,ind1)
            ENDIF
            IF (stg_trig_f_vol.gt.rmxstg(n,ind1,iseas)
     &      +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
              capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &         month)
            ELSE
              fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
              capac=fmultipl*discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &         ind1,month)*(stg_trig_f_vol-rmxstg(n,ind1,iseas))
            END IF
            capac_exc=capac
          ELSE
            IF (idiff_trigc_upstrmc .ne. 0) then
```

```
              stg_trig_f_vol = max(stg_trig_fc_ops(n,ind1),hw)
            ELSE
              stg_trig_f_vol = stg_trig_fc_ops(n,ind1)
            ENDIF
            IF (nint(wsdmnd).gt.0) THEN
              rminstg=stgintke(n,ind1,ns,imgt)
              IF (stg_trig_f_vol.gt.rmnstgs(n,ind1,iseas)
     &        +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                capac_exc=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &           ind1,month)
              ELSE IF (stg_trig_f_vol.gt.rmnstgs(n,ind1
     &        ,iseas)) THEN
                fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                capac_exc=fmultipl*discoef(n,ind1,ns,imgt)*
     &           frac_struc_cap(n,ind1,month)*(stg_trig_f_vol-rmnstgs
     &           (n,ind1,iseas))
              ELSE
                capac_exc=0.0
              END IF
              IF (hw.gt.rminstg+dpth_above_mn_cap_red(n,ind1,iseas))
     &        THEN
                capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &           month)
              ELSE IF (hw.gt.rminstg) THEN
                fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                capac=fmultipl*discoef(n,ind1,ns,imgt)*frac_struc_cap
     &           (n,ind1,month)*(hw-rminstg)
              ELSE
                capac=0.0
              END IF
            ELSE
              rminstg = rmnstgs(n,ind1,iseas)
              IF (stg_trig_f_vol.gt.rminstg+dpth_above_mn_cap_red
     &        (n,ind1,iseas)) THEN
                capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &           month)
              ELSE IF (stg_trig_f_vol.gt.rminstg) THEN
                fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                capac=fmultipl*discoef(n,ind1,ns,imgt)*frac_struc_cap
     &           (n,ind1,month)*(stg_trig_f_vol-rminstg)
              ELSE
                capac=0.0
              END IF
              capac_exc=capac
            END IF
C
          END IF
          IF (flnm(iflo).eq.strname_spec(12)
     &     .and.back_sp_pump_s9_opt.eq.'YES') THEN
            IF (iter.eq.0) THEN
              qu(n)=qu(n)+(1.00*rlvseep_to_cnl(il33)+1.00*
     &         rlvseep_to_cnl(il37))*86400.
            END IF
            IF (conv.eq.1.) THEN
              qu(il33)=qu(il33)-1.00*rlvseep_to_cnl(il33)*86400.
              qu(il37)=qu(il37)-1.00*rlvseep_to_cnl(il37)*86400.
            END IF
```

```
            END IF
c           if (n_gated_grav_outlets_fc(n) .gt. 0 .or. n_pump_outlets_fc
c    $      (n).eq. 0 .or. nout(n) .gt. 0) then
             IF (lpump(n,ind1,ns)) THEN
              if (imatch_struc_loc(n,ind1) .ne. 0) then
                  rminstg = rmin_stg_avol(n,ind1)
              else
                  rminstg = rmnstgs(n,ind1,iseas)
              endif
              IF (nint(wsdmnd).eq.0) THEN
                IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                  IF (trg_stg_hw.gt.rminstg
     &            +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    avvolq=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &               ind1,month)
                  ELSE IF (trg_stg_hw.gt.rminstg) THEN
                    fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                    avvolq=fmultipl*discoef(n,ind1,ns,imgt)*
     &               frac_struc_cap(n,ind1,month)*(trg_stg_hw-rminstg)
                  ELSE
                    avvolq=0.0
                  END IF
                ELSE
                  avvolq = (trg_stg_hw-rminstg)*cl(n)*width(n)
     &            + cgseep(n,rminstg,0,0.) + tovflc(n,rminstg
     &            ,0,0.,iter,1) + qu(n)
                  avvolq = amax1(avvolq,0.0)/86400.
                ENDIF
                avvolq_exc=avvolq
              ELSE
                if (imatch_struc_loc(n,ind1) .ne. 0) then
                   rminstg = rmin_stg_avol(n,ind1)
                else
                   rminstg = rmnstgs(n,ind1,iseas)
                endif
                IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                  IF (trg_stg_hw.gt.rminstg
     &            +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    avvolq_exc=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &               ind1,month)
                  ELSE IF (trg_stg_hw.gt.rminstg) THEN
                    fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                    avvolq_exc=fmultipl*discoef(n,ind1,ns,imgt)*
     &              frac_struc_cap(n,ind1,month)*(trg_stg_hw-rminstg)
                  ELSE
                    avvolq_exc=0.0
                  END IF
                ELSE
                  avvolq_exc = (trg_stg_hw-rminstg)*cl(n)*width(n)
     &            + cgseep(n,rminstg,0,0.) + tovflc(n,rminstg
     &            ,0,0.,iter,1) + qu(n)
                  avvolq_exc = amax1(avvolq_exc,0.0)/86400.
                ENDIF
                IF (hw.gt.stgintke(n,ind1,ns,imgt)
     &          +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                  avvolq=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &             ind1,month)
```

```
                ELSE IF (hw.gt.stgintke(n,ind1,ns,imgt)) THEN
                  fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                  avvolq=fmultipl*discoef(n,ind1,ns,imgt)*
     &             frac_struc_cap(n,ind1,month)*(hw-stgintke
     &             (n,ind1,ns,imgt))
                ELSE
                  avvolq=0.0
                END IF
              END IF
              capac=amin1(avvolq,capac)
              capac_exc=amin1(avvolq_exc,capac_exc)
             ELSE
              if (imatch_struc_loc(n,ind1) .ne. 0) then
                 rminstg = rmin_stg_avol(n,ind1)
              else
                 rminstg = rmnstgs(n,ind1,iseas)
              endif
              IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                IF (trg_stg_hw.gt.rminstg
     &          +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                  avvolq_exc=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &             ind1,month)
                ELSE IF (trg_stg_hw.gt.rminstg) THEN
                  fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                  avvolq_exc=fmultipl*discoef(n,ind1,ns,imgt)*
     &            frac_struc_cap(n,ind1,month)*(trg_stg_hw-rminstg)
                ELSE
                  avvolq_exc=0.0
                END IF
              ELSE
                avvolq_exc = (trg_stg_hw-rminstg)*cl(n)*width(n)
     &          + cgseep(n,rminstg,0,0.) + tovflc(n,rminstg
     &          ,0,0.,iter,1) + qu(n)
                avvolq_exc = amax1(avvolq_exc,0.0)/86400.
              ENDIF
              capac=amin1(avvolq_exc,capac)
              capac_exc=capac
             END IF
c           endif
            IF (conv.eq.1) lpump(n,ind1,ns)=.true.
            IF (conv.eq.1) THEN
              lpump(n,ind1,ns)=.not.(stg_trig_fc_ops(n,ind1).lt.
     &         rmnstgs(n,ind1,iseas).and. icount_bel_fc_thres(n,ind1
     &         ,ns) .gt. 5)
              IF (stg_trig_fc_ops(n,ind1) .lt. rmnstgs(n,ind1,iseas))
     &         icount_bel_fc_thres(n,ind1,ns) = icount_bel_fc_thres
     &         (n,ind1,ns) + 1
            END IF
          ELSE
            IF (nint(wsdmnd).gt.0) THEN
              rminstg=stgintke(n,ind1,ns,imgt)
              IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                IF (trg_stg_hw.gt.rmnstgs(n,ind1,iseas)
     &          +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                  avvolq_exc=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,
     &             ind1,month)
                ELSE IF (trg_stg_hw.gt.rmnstgs(n,ind1,iseas)) THEN
```

```
                  fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                  avvolq_exc=fmultipl*discoef(n,ind1,ns,imgt)*
     &            frac_struc_cap(n,ind1,month)*(trg_stg_hw
     &            - rmnstgs(n,ind1,iseas))
                ELSE
                  avvolq_exc=0.0
                END IF
              ELSE
                rminstg_f = rmnstgs(n,ind1,iseas)
                avvolq_exc = (trg_stg_hw-rminstg_f)*cl(n)*width(n)
     &          + cgseep(n,rminstg_f,0,0.) + tovflc(n,rminstg_f
     &          ,0,0.,iter,1) + qu(n)
                avvolq_exc = amax1(avvolq_exc,0.0)/86400.
              ENDIF
              IF (hw.gt.rminstg+dpth_above_mn_cap_red(n,ind1,iseas))
     &         THEN
                capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,
     &           month)
              ELSE IF (hw.gt.rminstg) THEN
                fmultipl = 1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                capac=fmultipl*discoef(n,ind1,ns,imgt)*frac_struc_cap
     &           (n,ind1,month)*(hw-rminstg)
              ELSE
                capac=0.0
              END IF
c             avvolq_exc=(trg_stg_hw-rmnstgs(n,ind1,iseas))*cl(n)
c    &         *width(n)+cgseep(n,rdn_stg_lim,0,0.)
c    &         +tovflc(n,rdn_stg_lim,0,0.,iter,1)+qu(n)
c             avvolq_exc=amax1(avvolq_exc,0.0)/86400.
c             avvolq=(hw-stgintke(n,ind1,ns,imgt))*cl(n)*width(n)+
c    &         cgseep(n,rdn_stg_lim_ws,0,0.)+tovflc(n,
c    &         rdn_stg_lim_ws,0,0.,iter,1)+qu(n)
c             avvolq=amax1(avvolq,0.0)/86400.
c             capac=amin1(avvolq,capac)
              capac_exc=amin1(avvolq_exc,capac)
            ELSE
              avvolq=0.0
              capac=0.0
              capac_exc=0.0
            END IF
            if (conv .eq. 1.) icount_bel_fc_thres(n,ind1,ns) =
     &      icount_bel_fc_thres(n,ind1,ns) + 1
          END IF
c         IF (conv.eq.1) THEN
c         IF (flnm(iflo).eq.'NNRFP ') print *,lpump(n,ind1,ns),
c    &    trg_stg_hw,rmnstgs(n,ind1,iseas),icount_bel_fc_thres(n,ind1
c    &    ,ns),'  NNRFP'
c    &    ,rmnstgs(n,ind1,iseas),lpump(n,ind1,ns),capac,capac_exc
c    &    ,stg_trig_fc_ops(n,ind1),rminstg,rmin_stg_avol(n,ind1)
c    &    ,qu(n)/86400.,month,iday,stg_trig_f_vol,flnm(iflo)
c         ENDIF
          IF (flnm(iflo).eq.'C9TC9R') THEN
            avvolq=avvolq-amin1(flow_to_c9r_frm_c11+
     &       tot_flw_to_c9r_frm_nlkblt+rnlkblt_to_s29,avvolq)
            avvolq_exc=avvolq_exc-amin1(flow_to_c9r_frm_c11+
     &       tot_flw_to_c9r_frm_nlkblt+rnlkblt_to_s29,avvolq_exc)
            capac=capac-amin1(flow_to_c9r_frm_c11+
```

```
     &       tot_flw_to_c9r_frm_nlkblt+rnlkblt_to_s29,capac)
             capac_exc=capac_exc-amin1(flow_to_c9r_frm_c11+
     &       tot_flw_to_c9r_frm_nlkblt+rnlkblt_to_s29,capac_exc)
             capac=amin1(capac,avvolq)
             capac_exc=amin1(capac_exc,avvolq_exc)
           END IF
           IF (flnm(iflo).eq.'C9TLBN') THEN
             avvolq=avvolq-amin1(rnlkblt_to_s29,avvolq)
             avvolq_exc=avvolq_exc-amin1(rnlkblt_to_s29,avvolq_exc)
             capac=capac-amin1(rnlkblt_to_s29,capac)
             capac_exc=capac_exc-amin1(rnlkblt_to_s29,capac_exc)
             capac=amin1(capac,avvolq)
             capac_exc=amin1(capac_exc,avvolq_exc)
           END IF
           IF (flnm(iflo).eq.'ACMRIN') THEN
             avvolq=avvolq-amin1(dischg_bld_acmres,avvolq)
             avvolq_exc=avvolq_exc-amin1(dischg_bld_acmres,avvolq_exc)
             capac=amin1(capac,avvolq)
             capac_exc=amin1(capac_exc,avvolq_exc)
           END IF
C
           IF (flnm(iflo).eq.'ACBC51') THEN
             avail_vol_locfc=amin1(avvolq_exc-qu(n)/86400,capac_exc)
             capac=capac+qu(n)/86400.
             capac_exc=capac_exc+qu(n)/86400.
           END IF
           capac=amin1(capac,capacp)
           capac_exc=amin1(capac_exc,capacp)
           capacfldc=capac
         ELSE IF (irating(n,ind1,ns,imgt).eq.3) THEN
           IF (nint(wsdmnd).gt.0) THEN
             tw=crmin(idncl,iseas)+stgdiff_canal_tw
           ELSE
             tw=rmnstgs(n,ind1,iseas)
           END IF
C
           IF (hw.gt.tw) THEN
             capac=discoef(n,ind1,ns,imgt)*frac_struc_cap(n,ind1,month)*
     &        (hw-tw)**pwr(n,ind1,ns,imgt)
           ELSE
             capac=0.0
           END IF
           IF (ides_cap_limit(n,ind1).eq.1) THEN
             capac=amin1(capac,strdcap(n,ind1,ns,imgt))
           END IF
           capacfldc=capac
           lpump(n,ind1,ns)=.false.
           IF (flnm(istr(n,ind1)).eq.strname_spec(67)) tot_grav_cap=
     &      capac+tot_grav_cap
         END IF
C
C  PUMP DOWNSTREAM NO MORE THAN CAN BE ACCEPTED WITHOUT AGGRAVATING
C  FLOOD CONDITIONS,IF APPLICABLE
C
c        if (flnm(iflo).eq.'S335 '.and. conv.eq.1.) print *,
c    &     tdqu_dn(idncl),dqu(l31s),dqu(l31w),dqu(ic4),dqu(isncre)
c    &     ,'  WSNEED H'
```

```
        IF (nint(tdqu_dn(idncl)).eq.0) THEN
c
c check if structure is constrained by remote downstream location
c
          istruc_on_off = 0
          DO kk = 1,no_struc_sing_on_off
              IF (n.eq.iups_cnl_no_on_off(kk).and.ind1.eq.
     &        istruc_no_on_off(kk)) THEN
                 IF (node_pos_on_off(kk) .gt. 0) THEN
                   IF (nodcr(icnl_no_on_off(kk)).gt.1) THEN
                     canl_stg_diff_frm_dnstrm = float(nodcr
     &               (icnl_no_on_off(kk))-node_pos_on_off(kk))
     &               /float(nodcr(icnl_no_on_off(kk))-1)
     &               *hdc(icnl_no_on_off(kk),iseas)
                   ELSE
                     canl_stg_diff_frm_dnstrm = 0.0
                   ENDIF
                   stg_trig_on_off(kk) = chdep(icnl_no_on_off(kk))
     &              + canl_stg_diff_frm_dnstrm
                 ELSE
                    stg_trig_on_off(kk) = h(igrid_loc_on_off(kk))
     &              + pond(igrid_loc_on_off(kk))
                 ENDIF
                 IF (stg_trig_on_off(kk) .lt. trig_stg_on_off(kk)
     &           -offset_on_off(kk))THEN
                     tvolmxdn=capacfldc
                 ELSEIF (stg_trig_on_off(kk).lt.trig_stg_on_off(kk))
     &           THEN
                     tvolmxdn=(1.0/offset_on_off(kk))*capacfldc
     &                *(trig_stg_on_off(kk)-stg_trig_on_off(kk))
                 ELSE
                     tvolmxdn=0.0
                 ENDIF
                 istruc_on_off = 1
                 GO TO 15
               ENDIF
          ENDDO

c structure NOT triggered by remote downstream location
c
          tvolmxdn=0.0
          IF (ndsclim(n,ind1).gt.0) THEN
            DO nc=1,ndsclim(n,ind1)
              IF (nodcr(icnum(n,ind1,nc)).gt.1) THEN
                stagediff=float(nodcr(icnum(n,ind1,nc))-nodepos_sd(n,
     &          ind1))/float(nodcr(icnum(n,ind1,nc))-1)*hdc(icnum(n,
     &          ind1,nc),iseas)
              ELSE
                stagediff=0.0
              ENDIF
              tdschgdn=0.0
              IF (into(icnum(n,ind1,nc),1).ge.0) THEN
                itide_down=0
                DO k=1,nout(icnum(n,ind1,nc))
                  icanal_down=into(icnum(n,ind1,nc),k)
                  IF (icanal_down.eq.0) THEN
                    itide_down=itide_down+1
```

```
                  twdph_down=get_tide(ycnlnode(icnum(n,ind1,nc),
     &             itide_down),jday,proj_sea_level_rise)
                  twdph_down=1.025*(twdph_down+2.0)-2.0
                  crelev_down=amax1(crel(icnum(n,ind1,nc),k),
     &             twdph_down)
                ELSE
                  crelev_down=crel(icnum(n,ind1,nc),k)
                END IF
                IF (cnlstg(n,ind1,nc)-stagediff.gt.crelev_down) THEN
                  IF (into(icnum(n,ind1,nc),1).eq.0) THEN
                    dschgdn=strcf(icnum(n,ind1,nc),k)*gwdth(icnum(n,
     &               ind1,nc),k)*(cnlstg(n,ind1,nc)-stagediff-
     &               crelev_down)*sqrt(cnlstg(n,ind1,nc)-stagediff-
     &               crelev_down)
                  ELSE
                    dschgdn=strcf(icnum(n,ind1,nc),k)*gwdth(icnum(n,
     &               ind1,nc),k)*sqrt(cnlstg(n,ind1,nc)-stagediff-
     &               crelev_down)
                  END IF
                  tdschgdn=tdschgdn+dschgdn*86400.
                END IF
              END DO
            END IF
            tot_pump=0.0
            DO k=1,nstrctr(icnum(n,ind1,nc))
              IF (nodcr(icnum(n,ind1,nc)).gt.1) then
                stagedff=float(nodcr(icnum(n,ind1,nc))-
     &          nodepos_su(icnum(n,ind1,nc),k))/float(nodcr(icnum(n,
     &          ind1,nc))-1)*hdc(icnum(n,ind1,nc),iseas)
              ELSE
                stagedff=0.0
              ENDIF
              stage_diff=stagediff-stagedff
              dschgdn=strout(cnlstg(n,ind1,nc)-stage_diff,icnum(n,
     &        ind1,nc),icnum(n,ind1,nc),k,0,crmin,volwtr,prevqu,rain,
     &        iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
              tdschgdn=tdschgdn+dschgdn*86400.
c             if (conv.eq.1 .and. flnm(iflo).eq.'S335  ') print *,
c    $         flnm(istr(ICNUM(N,IND1,NC),K)),DSCHGDN,
c    $         CNLSTG(N,IND1,NC)-stage_diff,month,iday
            END DO
            IF (icnum(n,ind1,nc).lt.n) THEN
              qinflwdn=prevqu(icnum(n,ind1,nc))
            ELSE
              qinflwdn=qu(icnum(n,ind1,nc))
              IF (flnm(iflo).eq.'S335  '.and. l29 .gt. l30) qinflwdn
     &        =qu(icnum(n,ind1,nc) )+s334_fc*86400.
            END IF
            avvspovdn=-cgseep(icnum(n,ind1,nc),cnlstg(n,ind1,nc)-
     &       stagediff,0,0.)-tovflc(icnum(n,ind1,nc),cnlstg(n,ind1,nc)
     &       -stagediff,0,0.,iter,2)-qinflwdn+tdschgdn+(cnlstg(n,ind1,
     &       nc)-stagediff-chdep(icnum(n,ind1,nc)))*cl(icnum(n,ind1,
     &       nc))*width(icnum(n,ind1,nc))
            avvolcdn=amax1(avvspovdn,0.0)/86400.
            tvolmxdn=tvolmxdn+avvolcdn
c            if (conv.eq.1. .and. cnm(n).eq.'L29 ') print *,CGSEEP
c    $         (ICNUM(N,IND1,NC),CNLSTG(N,IND1,NC)-stagediff,0,0.)
```

```
c    $           /86400.,TOVFLC(ICNUM(N,IND1,NC),CNLSTG(N,IND1,NC)
c    $           -stagediff,0,0.,ITER,2)/86400.,QINFLWDN/86400.,
c    $           TDSCHGDN/86400.,CNLSTG(N,IND1,NC),stagediff,'  S334 '
c    $           ,AVVOLCDN,TVOLMXDN,CAPAC,s334_fc,capacfldc
            END DO
            capac=amin1(capacfldc,tvolmxdn)
            capacfldc = capac
c           if (conv.eq.1. .and. flnm(iflo).eq.'S335  ')
c    &       print *,qu(l31nc)/86400.,s334_fc
c    &       ,qinflwdn/86400.,qu(n)/86400.,s337fc,'  S335  '
c    &       ,capacfldc,tvolmxdn,capac
          END IF
          if (istruc_on_off .eq. 0) go to 16
  15      capac=amin1(capacfldc,tvolmxdn)
c         if (conv.eq.1.)
c    &    print *,flnm(iflo),stg_trig_on_off(1),trig_stg_on_off(1),
c    &       capacfldc,tvolmxdn,'    S380R ',chdep(idncl),hdc(idncl
c    &       ,iseas),hw,tw,cnm(idncl)
          capacfldc = capac
  16      continue
        END IF
C
        IF (ipurpse(n,ind1,ns).eq.1.or.ipurpse(n,ind1,ns).eq.3) THEN
          capacws(imgt)=capac+capacws(imgt)
        END IF
        IF (ipurpse(n,ind1,ns).eq.2.or.ipurpse(n,ind1,ns).eq.3) THEN
          IF (irating(n,ind1,ns,imgt).eq.2) THEN
            capacfl(imgt,ns)=capac_exc
          ELSE
            capacfl(imgt,ns)=capacfldc
          END IF
        END IF
   20 CONTINUE
      RETURN
      END
```

canl_dep_struc_param_setup.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: canl_dep_struc_param_setup.F,v 1.12 2005/06/10 13:29:01 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/canl_dep_struc_param_setup.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE COMPUTES SETS UP THE BASIC PARAMETERS SUCH AS HEADWATER.
C   TAILWATER,DOWNSTREAM CONMSTRAINTS,VOLUME OF DOWNSTREAM WATER SUPPLY NEEDS
C   FOR COMPUTATION OF DISCHARGE THROUGH STRUCTURE.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE canl_dep_struc_param_setup (n, ind1, idncl, capacws,
     & capacfl, wsdmnd, imgt, logdisg, tdqu_dn, ndsglim, nodedlim,
     & fl_trig_constrt, tdmnd_orig, hws, tws, gravity_flow_struc, cout,
     & struc_name_meas, lpmps309, stgdiff_canal_tw,itideout,nstideout,
     & jday)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'et.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'opersched.inc'
```

```
      INCLUDE 'lokres.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'dualops.inc'

      DIMENSION capacfl(max_n_seas,max_n_substr), capacws(max_n_seas),
     & fl_trig_constrt(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
     & ndsglim(max_ncnls,max_nostr_cnl),
     & nodedlim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
     & tdqu_dn(max_ncnls)

      CHARACTER struc_name_meas(max_n_struc_meas)*6

      LOGICAL ldishg, logdisg, lpmps309

      DOUBLE PRECISION hws,tws
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      iflo=istr(n,ind1)
      IF (flnm(iflo).ne.strname_spec(1).and.flnm(iflo)
     &.ne.strname_spec(2).and.flnm(iflo).ne.strname_spec(3)
     &.and.runmode.eq.'SIMUL') kflo(iflo)=0.0
      idncl=idncnl(n,ind1)
C
      wsdmnd=0.0
      gravity_flow_struc=0.0
      DO i=1,no_ds_cnl_ws(n,ind1)
        IF (flnm(iflo).ne.strname_spec(75)) THEN
          wsdmnd=tdqu_dn(idncnl_ws(n,ind1,i))+wsdmnd
        ELSE
          IF (use_enp_ws_to_lec.eq.'TRUE ') THEN
            wsdmnd=ws_need_l31nc_orig+wsdmnd
          END IF
        END IF
      END DO
C     IF (runmode .eq. 'CALIB') THEN
      IF (cnm(n).eq.canl_names_spec(31)) THEN
        CALL match ('S334  ', 1, 6, struc_name_meas, nflpts, 6,
     &   imatch_s334)
        CALL match ('S336  ', 1, 6, struc_name_meas, nflpts, 6,
     &   imatch_s336)
        CALL match ('S338  ', 1, 6, struc_name_meas, nflpts, 6,
     &   imatch_s338)
        IF (ifix(kflo(ifls334)).gt.0.or.tdmnd_orig.gt.0.01) THEN
          IF (imatch_s334.ne.0.or.runmode.eq.'CALIB') THEN
            wsdmnd=wsdmnd-kflo(ifls334)
          END IF
          IF (imatch_s336.ne.0.or.runmode.eq.'CALIB') THEN
            wsdmnd=wsdmnd+kflo(ifls336)
          END IF
          IF (imatch_s338.ne.0.or.runmode.eq.'CALIB') THEN
            wsdmnd=wsdmnd+kflo(ifls338)
          END IF
          wsdmnd=amax1(wsdmnd,0.0)
        END IF
      END IF
C     ENDIF
```

```
C
      IF (dryseas) THEN
        iseas=1
      ELSE
        iseas=2
      END IF
C
      IF (day1) THEN
        DO ns=1,nsubstr(n,ind1)
          lpump(n,ind1,ns)=.false.
          lgrav(n,ind1,ns)=.false.
        END DO
      END IF
      idshg=0
      IF (ndsclim(n,ind1).gt.1) THEN
        DO 10 i=1,ndsclim(n,ind1)
          IF (chdep(icnum(n,ind1,i)).ge.cnlstg(n,ind1,i)) THEN
            idshg=idshg+1
          END IF
   10   CONTINUE
      END IF
      IF (ndsglim(n,ind1).gt.0) THEN
        DO i=1,ndsglim(n,ind1)
          deptgw=(ells(nodedlim(n,ind1,i))-h(nodedlim(n,ind1,i)))*
     &     s(nodedlim(n,ind1,i))
          IF (pond(nodedlim(n,ind1,i)).gt.deptgw) THEN
            dntrg_stg=pond(nodedlim(n,ind1,i))-deptgw+ells(nodedlim(n,
     &       ind1,i))
          ELSE
            dntrg_stg=(pond(nodedlim(n,ind1,i))-deptgw)/s(nodedlim(n,
     &       ind1,i))+ells(nodedlim(n,ind1,i))
          END IF
          IF (dntrg_stg.gt.fl_trig_constrt(n,ind1,i)) THEN
            idshg=idshg+1
          END IF
        END DO
      END IF
      ldishg=idshg.gt.0
C
      IF (ndsclim(n,ind1).gt.1) THEN
        IF (flnm(iflo).eq.strname_spec(13)) THEN
          logdisg=ldishg
        ELSE
          logdisg=.not.ldishg
        END IF
      ELSE IF (ndsglim(n,ind1).gt.0) THEN
        logdisg=.not.ldishg
      ELSE IF (flnm(iflo).eq.strname_spec(21)
     &.and. .not. lpmps309 .and. opt_l8_prop) THEN
        logdisg=.false.
      ELSE
        logdisg=.true.
      END IF
C
C
      IF (irceive(n,ind1).eq.1) THEN
        IF (idncl.le.nch) THEN
```

```
          IF (nodcr(idncl).gt.1) THEN
            stgdiff_canal_tw=float(nodcr(idncl)-nodepos_sd(n,ind1))/
     &      float(nodcr(idncl)-1)*hdc(idncl,iseas)
          ELSE
            stgdiff_canal_tw=0.0
          ENDIF
          tw=chdep(idncl)+stgdiff_canal_tw
        END IF
        IF (flnm(iflo).eq.strname_spec(37).or.flnm(iflo)
     &   .eq.strname_spec(21)) THEN
          IF (lok_split_option.eq.0) THEN
            tw=stagelo
          ELSE
            tw=amax1(stagelo_res,stagelo)
          END IF
        END IF
        IF (icell_no_sd(n,ind1) .gt. 0) THEN
           IF (lutyp(icell_no_sd(n,ind1)) .eq. 0) then
              nstideout = nstideout + 1
              twdph=get_tide(ycnlnode(n,nstideout+itideout),jday,
     &         proj_sea_level_rise)
              tw = 1.025*(twdph + 2.0) - 2.0
           ENDIF
        ENDIF
      ELSE IF (irceive(n,ind1).eq.2.or.irceive(n,ind1).eq.3) THEN
        tw=ells(nodrcv(n,ind1,1))+pond(nodrcv(n,ind1,1))
      END IF
      tws=tw
      if (nodcr(n).gt.1) then
        stgdiff_canal_hw=float(nodcr(n)-nodepos_su(n,ind1))/
     &  float(nodcr(n)-1)*hdc(n,iseas)
      else
        stgdiff_canal_hw=0.0
      endif
      hws=chdep(n)+stgdiff_canal_hw
C
      IF (ncoeff(n,ind1).eq.1) THEN
        imgt=1
      ELSE
        imgt=iseas
      END IF
      cout=0.0
C
      DO jj=1,ncoeff(n,ind1)
        capacws(jj)=0.0
        DO kk=1,nsubstr(n,ind1)
          capacfl(jj,kk)=0.0
        END DO
      END DO
      RETURN
      END
```

ccaloos.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: ccaloos.inc,v 1.2 2003/07/03 20:07:56 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ccaloos.inc,v $
C       $Name: rel-5-5 $
C                                                              -
C-----------------------------------------------------------------------
      common /ccaloos/ ealc43ro,alc43,eflc43,eqs79,eqs77
     &      ,alcae0,c43ro0,caedmd0,s77cap0,s79cap0,s77cap,s79cap
     &      ,c43est_met_by_lok,eallokres1,c43dmd_met_by_lokres
```

chnlf.F

```
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: chnlf.F,v 1.35 2005/06/10 13:29:01 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/chnlf.F,v $
C   $Name: rel-5-5 $
C
C
C   PURPOSE:
C       TO CALCULATE DOWNSTREAM STAGE, SEEPAGE ALONG CANAL, DOWNSTREAM OUTFLOW,
C       AND OVERLAND INFLOW GIVEN UPSTREAM INFLOS AND DOWNSTREAM STAGE AT THE
END
C       OF THE PREVIOUS DAY. HEAD DROPS FOR THE MAJOR PROJECT CANALS (NAMES AND
C       PERTINENT INFORMATION ARE INPUT) IN EAST COAST SERVICE AREAS ARE
C       CALCULATED ON DAILY BASIS.
C
C
C   VARIABLE DEFINITIONS:
C           WIDTH             - WIDTH OF REACH. (FT)
C           CL                - CHANNEL LENGTH. (FT)
C           NCH               - NUMBER OF CHANNELS.
C           NODCR             - NUMBER OF NODES IN THE CHANNEL
C           XCN,YCN           - DEFINES CHANNEL NODE POINTS
C           INTO              - NUMBER OF OUTFLOWS FOR EACH REACH
C           LCNB              - INDICATES WATER BASIN CANAL RUNS IN
C           CREL              - DOWNSTREAM CREST ELEVATION (FT)
C           GWDTH             - DOWNSTREAM GATE WIDTH (FT)
C           NOUT              - NUMBER OF OUTFLOWS FOR EACH REACH
C           COUT              - DAILY AND MONTHLY DOWNSTREAM OUTFLOW
C           CGINF,CGINFM      - DAILY AND MONTHLY TOTAL CANAL
C                               SEEPAGE.
C           QU,QUM            - DAILY AND MONTHLY UPSTREAM INFLO
C           VINM              - MONTHLY UPSTREAM AND OVERLAND INFLO
C           DSWLM             - MONTHLY CHANGE IN STORAGE OF CANAL
C           SWL               - SURFACE WATER LEVEL ESTIMATION
C                               USED IN ITERATION PROCESS.(FT)
C           SWLAST            - SURFACE WATER LEVEL FROM  PREVIOUS
C                               ITERATION (FT)
```

```
C            ADEP              - CORRECTION IN SWL FOR NEXT
C                                ITERATION.(FT)
C            DISTNC            - LENGTH OF CANAL REACH AT A NODE
C            VOLIN             - CALCULATED INTERFLOW WITH SURFACE
C                                WATER FOR THE ENTIRE REACH(FT3)
C            CGINF             - CALCULATED INTERFLOW WITH
C                                THE GROUNDWATER AT EACH NODE
C            CGSEEP            - CALCULATED INTERFLOW WITH THE GW
C                                SUMED FOR THE ENTIRE REACH.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE chnlf (idymax, rain, avol, crmin, rfls155a, jday,
     & volwtr, prevqu, finstor, tinflow, outflow_from_3a, head_offset,
     & total_flow_to_mcnl, rlist_lvname, stagelo_beg, stgref,
     & finstor_res, rmflws12, total_flow_s12, s12_reg_total,
     & s12_env_total, rmflws333, temp_volwtr, tempf_volwtr,
     & rloc_seep_for_ws, rlocal_ws_to_e1, rlocal_ws_to_e2,
     & rlocal_ws_to_e3, rlocal_ws_to_es, rlocal_ws_to_ese,
     & ws_to_l8_via_s352, convey_canal_names, nconvey_total, pctmet,
     & temp_weircff, tempf_weircff, rloc_excess_for_ws,
     & rloc_excss_to_canl_next_step, dischg_to_bay_s29_frm_c11r,
     & s333_reg_lec, tot_cnl_dmd_asr, tot_flw_to_c9r_frm_nlkblt,
     & flow_to_cpbres_frm_e1, flow_to_c9r_frm_c9, flow_to_lbn_frm_c11,
     & rnlkblt_to_s29, flow_to_ca1_frm_acmebsn, dellec,
     & s333_reg_lec_abve_sch, s333_reg_lec_bel_sch, total_small_afact,
     & ndsglim, nodedlim, fl_trig_constrt, struc_name_meas, node_wd,
     & rmean_cnl_bot_elev, flow_resistvty_coef, iwuppflw_for_slope,
     & iwdnpflw_for_slope, isuppflw_for_slope, isdnpflw_for_slope,
     & rf_drawdwn_dph, c304_out_to_marsh, hdrop_fact, hdrop_max,
     & rlvseep_to_cnl, s356_to_cell, s356_to_l29,
     & is334_open_flg, frac_avail_cap_s333,frac_avail_cap_s334,
     & weir_name, s333s334rg_offset, iroute_to_wca3a_frm_holy,
     & stage_g3273,HWSEEP_F_L30,HDSEEP_F_L30,ilv_f_l30,no_of_l30spnod,
     & icol_f_l30,irow_f_l30,rlvseep_to_l30,ipos_lv_l30,daily_seep,
     & seepmon, tot_avl_l8_bpit_res_m1q, flow_to_c9r_frm_c11,hdrop_min,
     & add_dmndagl8,n_gated_grav_outlets_fc,n_pump_outlets_fc,
     & rmax_dschg_coef,isupp_at_ups_end,out_flow_east,out_flow_east0,
     & iroute_acme_fc_to_sta1e_opt,i_canal_rf_chg_ops,
     & frac_s319_sta1ew,frac_s319_sta1ee,iroute_l8rnf_to_sta_opt)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'simcstgs.inc'
```

```
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'dualops.inc'
      INCLUDE 'cnlneeds.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION avol(max_ncnls), canl_rf_avg(max_ncnls),
     & capacfl(max_n_seas,max_n_substr),daily_seep(max_n_lvsp_loc),
     & capacws(max_n_seas), cginfy(max_ncnls), cmn(max_ncnls),
     & cmx(max_ncnls), couty(max_ncnls),
     & crel3(max_ncnls,max_n_weir_outf), crmin(max_ncnls,max_n_seas),
     & dischg_for_slope_in(max_n_cnls_w_var_slpe,2,3), dswly(max_ncnls),
     & etout(max_ncnls), etoutm(max_ncnls), etouty(max_ncnls),
     & fl_trig_constrt(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
     & flow_resistvty_coef(max_n_cnls_w_var_slpe),
     & hdrop_fact(max_n_cnls_w_var_slpe),seepmon(max_n_lvsp_loc),
     & hdrop_max(max_n_cnls_w_var_slpe),rmax_dschg_coef(max_ncnls),
     & hdrop_min(max_n_cnls_w_var_slpe),
     & head_offset(max_ncnls),ilgth(max_ncnlloc),
     & isdnpflw_for_slope(max_ncnls,max_nostr_cnl),
     & isuppflw_for_slope(max_ncnls,max_nostr_cnl),
     & iwdnpflw_for_slope(max_ncnls,max_n_weir_outf),
     & iwuppflw_for_slope(max_ncnls,max_n_weir_outf),
     & ndsglim(max_ncnls,max_nostr_cnl),
     & node_wd(max_ncnls,max_n_weir_outf), nodec(max_ncnlloc),
     & nodedlim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
     & outflow_from_3a(max_n_conv_cnl_wca),
     & outflow_from_3a_temp(max_n_conv_cnl_wca), prevqu(max_ncnls),
     & quy(max_ncnls), rain(max_ncells),
     & rf_drawdwn_dph(max_ncnls,2,max_n_seas),
     & rf_in_canal(max_ncnls,15), rfin(max_ncnls), rfinm(max_ncnls),
     & rfiny(max_ncnls), rloc_excess_for_ws(max_n_serv_area),
     & rloc_excss_to_canl_next_step(max_ncnls),
     & rloc_seep_for_ws(max_n_serv_area), rlvseep_to_cnl(max_ncnls),
     & rmean_cnl_bot_elev(max_n_cnls_w_var_slpe),
     & rmnstgs3(max_ncnls,max_nostr_cnl,max_n_seas),
     & rmxstg3(max_ncnls,max_nostr_cnl,max_n_seas),
     & struc_flow_f_dss(max_nostr_cnl),
     & weir_struc_f_dss(max_n_weir_outf),
     & adj_for_error_weir(max_n_weir_outf),
     & adj_for_error_struc(max_nostr_cnl),
     & rnflwin(max_ncnls,max_n_weir_outf), temp_volwtr(max_n_res),
     & temp_weircff(max_n_res,max_nostr_res), tempf_volwtr(max_n_res),
     & tempf_weircff(max_n_res,max_nostr_res),
     & tinflow(max_n_wcas,max_n_conv_cnl_wca), tivolwtr(max_n_res),
     & total_small_afact(max_ncells), viny(max_ncnls),
     & volwtr(max_n_res), wratio(max_ncnls),
     & n_gated_grav_outlets_fc(max_ncnls),n_pump_outlets_fc(max_ncnls),
     & ipump_true_e_match(max_nostr_cnl),i_canal_rf_chg_ops(max_ncnls),
     & igrav_true_e_match(max_nostr_cnl),
     & dischg_for_slope_max(max_n_cnls_w_var_slpe,2,3),
     & isupp_at_ups_end(max_ncnls,max_nostr_cnl)

c      DIMENSION i_str_rec_idx(max_nostr_cnl),
```

```
c     & i_weir_str_idx(max_n_weir_outf)

      DIMENSION HWSEEP_F_L30(10),HDSEEP_F_L30(10),ilv_f_l30(10),
     & icol_f_l30(10),irow_f_l30(10),ipos_lv_l30(10)

      CHARACTER convey_canal_names(max_n_tot_conv_cnl_wcas)*5,
     & rlist_lvname(max_no_lvsp)*5, struc_name_meas(max_n_struc_meas)*6,
     & weir_name(max_ncnls,max_n_weir_outf)*6

      LOGICAL logdisg, lpmps309, lflwth, l334rg, lwsdel

      REAL mean_land_surf_along_canal

      DOUBLE PRECISION fac,crmin,swl_land_surf,hw,tw,hws,tws
     & ,stgdiff_canal_down,floor,hdrop_prev,stgdiff_canal_ups
     & ,fac_headdrop,HDC_IT,av_cstor,cstor_def,stgdn_sp
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      step=float(nstep)

      if (day1) then
         lflwth = .FALSE.
         l334rg = .FALSE.
         lwsdel = .FALSE.
         lpmps309 = .FALSE.
         avvol_dns334 = 0.0
         add_s333_fc_s334 = 0.0
         do n = 1,nch
           head_offset(n)=0.0
         enddo
      endif

      IF (iday.ne.1) GO TO 20
      DO 10 n=1,nch
        coutm(n)=0.0
        cginfm(n)=0.0
        qum(n)=0.0
        vinm(n)=0.0
        dswlm(n)=0.0
        rfinm(n)=0.0
        etoutm(n)=0.0
        cmx(n)=-100.
        cmn(n)=100.
   10 CONTINUE
      sum=0.0
!-----------------
!  INITIALIZE VARIABLES
!-----------------
   20 npts=0
      vol_inj_to_c51asr=0.0
      vol_inj_to_hlsbasr=0.0
      vol_inj_to_nnrcasr=0.0
      vol_inj_to_l31asr=0.0
      rlvseep_to_l30 = 0.0
      drainc51w=0.0
      drainc51=0.0
      drainc11w=0.0
```

```
      drainhil=0.0
      s12_env_total=0.0
      s12_reg_total=0.0
      total_flow_to_mcnl=0.0
      ihly_out_count=0
      nmdays_rainavg = nmdays_rainavg + 1
      IF (nmdays_rainavg .gt.14) nmdays_rainavg =14
!
      DO i=1,n_intcnl(iwca3a)
        outflow_from_3a_temp(i)=outflow_from_3a(i)
      END DO
!
!  INITIALIZE SURFACE WATER TO CANAL INTERACTION
!
      DO node=1,max_ncells
        ovlflo_to_canal(node)=0.0
        seep_to_canal(node)=0.0
        canal_evap(node)=0.0
      END DO
!
c
c  set up operations for canals that will be drawn down
c  during appropriate periods
c
      jstr1=jstr
      IF (ndays_prestorm_drawdown.gt.0) THEN
        IF (nstorm_day(ndays_sim_sd).eq.1) THEN
          DO id=1,idrwn
            IF (struc_name_drawdown(id).ne.'WIER ') THEN
              rmxstg(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &         iseas)=rmxstg_drawdown(id)
              rmnstgs(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &         iseas)=rmnstg_drawdown(id)
              DO j=1,NDSCLIM(icanal_no_drawdown(id),
     &        istruc_no_drawdown(id))
                CNLSTG(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &          j) = tw_max_struc_inflw_drwdwn(id,j)
              END DO
            ELSE
              DO kk=1,nout(icanal_no_drawdown(id))
                crel(icanal_no_drawdown(id),kk)=rmxstg_drawdown(id)
              ENDDO
            END IF
          END DO
        ELSE
          DO id=1,idrwn
            IF (struc_name_drawdown(id).ne.'WIER ') THEN
              rmxstg(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &         iseas)=rmxstg2(icanal_no_drawdown(id),
     &         istruc_no_drawdown(id),iseas)
              rmnstgs(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &         iseas)=rmnstg2(icanal_no_drawdown(id),
     &         istruc_no_drawdown(id),iseas)
              DO j=1,NDSCLIM(icanal_no_drawdown(id),
     &         istruc_no_drawdown(id))
                CNLSTG(icanal_no_drawdown(id),istruc_no_drawdown(id),
     &          j) = CNLSTG2(icanal_no_drawdown(id),
```

```
     &           istruc_no_drawdown(id),j)
               ENDDO
            ELSE
              DO kk = 1,nout(icanal_no_drawdown(id))
                crel(icanal_no_drawdown(id),kk)
     &           = crel2(icanal_no_drawdown(id),kk)
              ENDDO
            END IF
          END DO
        END IF
!      do id = 1,idrwn
!      if (struc_name_drawdown(id) .ne. 'WIER ') then
!      print *,rmxstg(icanal_no_drawdown(id),istruc_no_drawdown(id)
!    $   ,iseas),rmnstgs(icanal_no_drawdown(id),istruc_no_drawdown(id)
!    $   ,iseas),nstorm_day(ndays_sim_sd),month,iday,ifyr+iyear-1
!      endif
!      enddo
      END IF
c
c  incorporate dual operations for special cases
c
      IF (nstorm_day(ndays_sim_sd).ne.1) THEN
         IF (nint(s337fc) .gt. 0 .or. nint(s333_reg_lec) .gt. 0) THEN
          DO id=1,ndual_canal_ops(1)
              IF (struc_name_dual_ops(id,1).ne.'WIER ') THEN
                rmxstg(icanal_no_dualops(id,1),istruc_no_dualops(id,1),
     &           iseas)=rmxstg_dual_ops(id,1)
                rmnstgs(icanal_no_dualops(id,1),istruc_no_dualops(id,1),
     &           iseas)=rmnstg_dual_ops(id,1)
              ELSE
                DO kk=1,nout(icanal_no_dualops(id,1))
                  crel(icanal_no_dualops(id,1),kk)=rmxstg_dual_ops(id,1)
                ENDDO
              END IF
          END DO
        ELSE
          DO id=1,ndual_canal_ops(1)
            IF (struc_name_dual_ops(id,1).ne.'WIER ') THEN
              rmxstg(icanal_no_dualops(id,1),istruc_no_dualops(id,1),
     &         iseas)=rmxstg2(icanal_no_dualops(id,1),
     &         istruc_no_dualops(id,1),iseas)
              rmnstgs(icanal_no_dualops(id,1),istruc_no_dualops(id,1),
     &         iseas)=rmnstg2(icanal_no_dualops(id,1),
     &         istruc_no_dualops(id,1),iseas)
            ELSE
              DO kk = 1,nout(icanal_no_dualops(id,1))
                crel(icanal_no_dualops(id,1),kk)
     &           = crel2(icanal_no_dualops(id,1),kk)
              ENDDO
            END IF
          END DO
        END IF
         IF (nint(s333_reg_lec) .gt. 0) THEN
          DO id=1,ndual_canal_ops(2)
            IF (struc_name_dual_ops(id,2).ne.'WIER ') THEN
              rmxstg(icanal_no_dualops(id,2),istruc_no_dualops(id,2),
     &         iseas)=rmxstg_dual_ops(id,2)
```

```
              rmnstgs(icanal_no_dualops(id,2),istruc_no_dualops(id,2),
     &         iseas)=rmnstg_dual_ops(id,2)
            ELSE
              DO kk=1,nout(icanal_no_dualops(id,2))
                crel(icanal_no_dualops(id,2),kk)=rmxstg_dual_ops(id,2)
              ENDDO
            END IF
          END DO
        ELSE
          DO id=1,ndual_canal_ops(2)
            IF (struc_name_dual_ops(id,2).ne.'WIER ') THEN
              rmxstg(icanal_no_dualops(id,2),istruc_no_dualops(id,2),
     &         iseas)=rmxstg2(icanal_no_dualops(id,2),
     &         istruc_no_dualops(id,2),iseas)
              rmnstgs(icanal_no_dualops(id,2),istruc_no_dualops(id,2),
     &         iseas)=rmnstg2(icanal_no_dualops(id,2),
     &         istruc_no_dualops(id,2),iseas)
            ELSE
              DO kk = 1,nout(icanal_no_dualops(id,2))
                crel(icanal_no_dualops(id,2),kk)
     &           = crel2(icanal_no_dualops(id,2),kk)
              ENDDO
            END IF
          END DO
        END IF
c        IF (nint(s337fc) .gt. 0) THEN
c         DO id=1,ndual_canal_ops(3)
c           IF (struc_name_dual_ops(id,3).ne.'WIER ') THEN
c             rmxstg(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
c    &         iseas)=rmxstg_dual_ops(id,3)
c             rmnstgs(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
c    &         iseas)=rmnstg_dual_ops(id,3)
c           ELSE
c             DO kk=1,nout(icanal_no_dualops(id,3))
c               crel(icanal_no_dualops(id,3),kk)=rmxstg_dual_ops(id,3)
c             ENDDO
c           END IF
c         END DO
c        ELSE
c         DO id=1,ndual_canal_ops(3)
c           IF (struc_name_dual_ops(id,3).ne.'WIER ') THEN
c             rmxstg(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
c    &         iseas)=rmxstg2(icanal_no_dualops(id,3),
c    &         istruc_no_dualops(id,3),iseas)
c             rmnstgs(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
c    &         iseas)=rmnstg2(icanal_no_dualops(id,3),
c    &         istruc_no_dualops(id,3),iseas)
c           ELSE
c             DO kk = 1,nout(icanal_no_dualops(id,3))
c               crel(icanal_no_dualops(id,3),kk)
c    &           = crel2(icanal_no_dualops(id,3),kk)
c             ENDDO
c           END IF
c         END DO
c       END IF
      END IF
      DO n=1,nch
```

```
        tdqu_dn(n)=dqu(n)
        rloc_excss_to_canl_next_step(n)=0.0
        i_canal_rf_chg_ops(n) = 0
      END DO
!-----------------
      DO 130 n=1,nch
!
!  INITIALIZE VARIABLES
!

        carea=cl(n)*width(n)
        qud=0.0
        tovlf=0.0
        rfin(n)=0.0
        etout(n)=0.0
        do is = 1,nstrctr(n)
           do ir = 1,nres(n,is)
              nstrtores(irsrcvr(n,is,ir)) = 0
           enddo
        enddo
        dplast=chdep(n)+hdc(n,iseas)/2.0
!
        poutd=0.0
        pout=0.0
        dsum=0.0
        if (cnm(n) .eq. 'L30  ') qutemp = qu(n)
        IF (day1 .OR. nmdays_rainavg .EQ. 14) THEN
             canl_rf_avg(n) = 0.0
        ENDIF
!        TSTOR = 0.0
!
        mean_land_surf_along_canal=0.0
        DO 30 ic=1,nodcr(n)
          npts=npts+1
          nodec(ic)=nodecn(npts)/10
          ilgth(ic)=mod(nodecn(npts),10)
          rf=rain(nodec(ic))/(12.*step)
          etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
          rfin(n)=rfin(n)+rf*distnc(ilgth(ic))*width(n)
          etout(n)=etout(n)+etcn*distnc(ilgth(ic))*width(n)
          canal_evap(nodec(ic))=etcn*distnc(ilgth(ic))*width(n)/1000.
          rmonthly_canal_evap(nodec(ic))=rmonthly_canal_evap(nodec(ic))+
     &     canal_evap(nodec(ic))
          dsum=dsum+distnc(ilgth(ic))
          mean_land_surf_along_canal=mean_land_surf_along_canal+
     &     ells(nodec(ic))/nodcr(n)
   30   CONTINUE
c
        IF (nint(s337fc) .gt. 0) THEN
          DO id=1,ndual_canal_ops(3)
            IF (n .eq.icanal_no_dualops(id,3)) THEN
              IF (struc_name_dual_ops(id,3).ne.'WIER ') THEN
                rmxstg(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
     &          iseas)=rmxstg_dual_ops(id,3)
                rmnstgs(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
     &          iseas)=rmnstg_dual_ops(id,3)
              ELSE
```

```
          DO kk=1,nout(icanal_no_dualops(id,3))
            crel(icanal_no_dualops(id,3),kk)=rmxstg_dual_ops(id,3)
          ENDDO
        END IF
      ENDIF
    END DO
  ELSE
    DO id=1,ndual_canal_ops(3)
      IF (n .eq.icanal_no_dualops(id,3)) THEN
        IF (struc_name_dual_ops(id,3).ne.'WIER ') THEN
          rmxstg(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
&          iseas)=rmxstg2(icanal_no_dualops(id,3),
&          istruc_no_dualops(id,3),iseas)
          rmnstgs(icanal_no_dualops(id,3),istruc_no_dualops(id,3),
&          iseas)=rmnstg2(icanal_no_dualops(id,3),
&          istruc_no_dualops(id,3),iseas)
        ELSE
          DO kk = 1,nout(icanal_no_dualops(id,3))
            crel(icanal_no_dualops(id,3),kk)
&            = crel2(icanal_no_dualops(id,3),kk)
          ENDDO
        END IF
      ENDIF
    END DO
  END IF
  IF (ivary_slope_daily_opt(n).eq.1) then
    IF (ndays_sim .GE. 14) THEN
       DO idr = 1,nmdays_rainavg-1
          IF (ndays_sim .GT. 14) THEN
             rf_in_canal(n,idr) = rf_in_canal(n,idr+1)
          ENDIF
          canl_rf_avg(n) = canl_rf_avg(n) + rf_in_canal(n,idr)
&         /nmdays_rainavg
       ENDDO
       rf_in_canal(n,nmdays_rainavg) =rfin(n)/(width(n)*cl(n))*12.
       canl_rf_avg(n) =  canl_rf_avg(n) + rf_in_canal(n
&      ,nmdays_rainavg)/nmdays_rainavg
    ELSE
       rf_in_canal(n,nmdays_rainavg) =rfin(n)/(width(n)*cl(n))*12.
       canl_rf_avg(n) = rf_in_canal(n,nmdays_rainavg)
    ENDIF
    icanal_drawdwn = 0
    do id = 1,idrwn
      IF (n.eq. icanal_no_drawdown(id)) icanal_drawdwn = 1
    enddo
    IF (ndays_prestorm_drawdown .eq. 0 .OR. nstorm_day
&   (ndays_sim_sd).eq. 0 .OR. icanal_drawdwn .eq. 0 ) THEN
        DO jj=1,nout(n)
          IF (canl_rf_avg(n) .gt. 5.50/14.) then
            crel3(n,jj) = crel2(n,jj) - rf_drawdwn_dph(n,2,iseas)
            i_canal_rf_chg_ops(n) = 1
          elseif (canl_rf_avg(n) .ge. 3.80/14.) then
            crel3(n,jj) = crel2(n,jj) - rf_drawdwn_dph(n,1,iseas)
            i_canal_rf_chg_ops(n) = 1
          else
            crel3(n,jj) = crel2(n,jj)
            i_canal_rf_chg_ops(n) = 0
```

```
            endif
            crel(n,jj) = crel3(n,jj)
          ENDDO
          DO jj = 1,nstrctr(n)
            itrg_loc_diff = 0
            itrg_indx = 1
            DO kk = 1,nsfc_w_chg_trg
               IF (n.eq. icnl_no_upstrm_trg_ops(kk)
&              .and. jj .eq.istruc_no_chg_trg_ops(kk)) THEN
                  itrg_loc_diff = 1
                  jjk = kk
                  IF (istruc_no_alt_trg_ops(kk) .gt. 0) THEN
                     IF (nint(kflo(istruc_no_alt_trg_ops(kk)))
&                    .eq. 0) THEN
                        itrg_indx = 2
                     ELSE
                        itrg_indx = 1
                     ENDIF
                  ELSE
                   itrg_indx = 1
                  ENDIF
               ENDIF
            ENDDO
            IF (((isuppflw_for_slope(n,jj).eq.2 .or. (
&           isuppflw_for_slope(n,jj).eq.1 .and. isupp_at_ups_end
&           (n,jj).eq.1)) .and. itrg_loc_diff.eq.0) .or.
&           (itrg_loc_diff .eq. 1 .and. (isuppflw_for_slope(n,jj)
&           .eq.2 .or. (isuppflw_for_slope(n,jj).eq.1 .and.
&           isupp_at_ups_end(n,jj).eq.1)) .and.
&           itrg_loc_ops_cnl(jjk,itrg_indx).eq.1)) THEN
               rmxstg3(n,jj,iseas) = rmxstg2(n,jj,iseas)
               rmnstgs3(n,jj,iseas) = rmnstg2(n,jj,iseas)
               if (i_canal_rf_chg_ops(n) .eq. 0) THEN
                 i_canal_rf_chg_ops(n) = 0
               endif
            ELSE
              IF (canl_rf_avg(n) .gt. 5.50/14.) then
                rmxstg3(n,jj,iseas) = rmxstg2(n,jj,iseas)
&               - rf_drawdwn_dph(n,2,iseas)
                rmnstgs3(n,jj,iseas) = rmnstg2(n,jj,iseas)
&               - rf_drawdwn_dph(n,2,iseas)
                i_canal_rf_chg_ops(n) = 1
              ELSEIF(canl_rf_avg(n) .ge. 3.80/14.) then
                rmxstg3(n,jj,iseas) = rmxstg2(n,jj,iseas)
&               - rf_drawdwn_dph(n,1,iseas)
                rmnstgs3(n,jj,iseas) = rmnstg2(n,jj,iseas)
&               - rf_drawdwn_dph(n,1,iseas)
                i_canal_rf_chg_ops(n) = 1
              ELSE
                rmxstg3(n,jj,iseas) = rmxstg2(n,jj,iseas)
                rmnstgs3(n,jj,iseas) = rmnstg2(n,jj,iseas)
                i_canal_rf_chg_ops(n) = 0
              ENDIF
            ENDIF
            rmxstg(n,jj,iseas) = rmxstg3(n,jj,iseas)
            rmnstgs(n,jj,iseas) = rmnstgs3(n,jj,iseas)
          ENDDO
```

```
          ENDIF
        ENDIF
        do istruc = 1,nstrctr(n)
           if (no_match_loc(istruc) .gt. 0) THEN
              rminstg_match = 9999999.
              do k = 1,no_match_loc(istruc)
                 istrucm = imatch_struc_no(istruc,k)
                 IF (ncoeff(n,istrucm).eq.1) THEN
                   imgte=1
                 ELSE
                   imgte=iseas
                 END IF
                 do ns = 1,nsubstr(n,istrucm)
                   if (irating(n,istrucm,ns,imgte).eq.2
     $             .and. ipurpse(n,istrucm,ns) .ne. 1) then
                     rminstg_match_t = rmnstgs(n,istrucm,iseas)
                     if (rminstg_match_t .lt. rminstg_match) then
                       rminstg_match = rminstg_match_t
                       istrucm_min = istrucm
                     endif
                   endif
                 enddo
              enddo
              do k = 1,no_match_loc(istruc)
                 istrucm = imatch_struc_no(istruc,k)
                 rmin_stg_avol(n,istrucm) = rminstg_match
              enddo
           endif
        enddo
!
        swl_land_surf=mean_land_surf_along_canal-hdc(n,iseas)/2.0
        canal_stor=(swl_land_surf-chdep(n))*cl(n)*width(n)
        tstor=canal_stor-cgseep(n,swl_land_surf,0,0.)-abs(tovflc(n,
     &   swl_land_surf,0,0.,0,1))-rfin(n)+etout(n)
        tstor=amax1(tstor,0.0)
!
        IF (nout(n).gt.0) THEN
          IF (into(n,1).ge.0) THEN
            DO jj=1,nout(n)
              IF (swl_land_surf.gt.crel(n,jj)) THEN
                if (into(n,1).eq.0 .and. weir_name(n,jj).ne.'G93  ')
     &          then
                  pout=strcf(n,jj)*86400.*gwdth(n,jj)*(swl_land_surf
     &            -crel(n,jj))*sqrt(swl_land_surf-crel(n,jj))
                else
                  pout=strcf(n,jj)*86400.*gwdth(n,jj)*sqrt
     &            (swl_land_surf-crel(n,jj))
                endif
                poutd=poutd+pout
              END IF
            END DO
          END IF
        END IF
        tot_pump=0.0
        DO is=1,nstrctr(n)
          poutd=poutd+strout(swl_land_surf,n,n,is,0,crmin,volwtr,prevqu,
     &     rain,0,tot_pump,jday,tempf_volwtr,tempf_weircff,0)*86400.
```

```
        END DO
!
        tdmnd_dwns=0.0
        IF (nstrctr(n).gt.0) THEN
          DO in=1,nstrctr(n)
!
            IF (flnm(istr(n,in)).eq.'S332  '.and.opt_flow_to_ts.eq.'TSMI
     &NDL') THEN
              adddmnd=vol332(month)
            ELSE IF (flnm(istr(n,in)).eq.'S332  '
     &       .and.opt_flow_to_ts.ne.'TSMINDL'.and.opt_flow_to_ts.ne.'TSR
     &FPLN') THEN
              adddmnd=vol332do(month)
            ELSE IF (flnm(istr(n,in)).eq.'LXSLWS'.and.lwslo(iloxah))
     &       THEN
              adddmnd=discoef(n,in,1,1)
            ELSE IF (flnm(istr(n,in)).eq.'G92TRV') THEN
              adddmnd=strdcap(n,in,1,1)
            ELSE
              adddmnd=0.0
            END IF
!
            DO i=1,no_ds_cnl_ws(n,in)
              IF (ipurpse(n,in,1).ne.2) THEN
                IF (flnm(istr(n,in)).eq.'S334  ') THEN
                  ds_wsneed=ws_need_l31nc_orig
                ELSE
                  ds_wsneed=tdqu_dn(idncnl_ws(n,in,i))
                END IF
                tdmnd_dwns=tdmnd_dwns+ds_wsneed
              END IF
            END DO
            tdmnd_dwns=tdmnd_dwns+adddmnd
          END DO
        END IF
        IF ((qu(n)-poutd).gt.0.0.and.lcnb(n).ne.88.and.nint(tdmnd_dwns)
     &   .eq.0.and.((lcnb(n).ne.0.and.cnm(n).ne.canl_names_spec(42))
     &   .or.(cnm(n).eq.canl_names_spec(42).and.dqu(n).le.vol18c(month)/
     &   86400.+0.1)).and.lcnb(n).ne.11.and.cnm(n)
     &   .ne.canl_names_spec(30).and.cnm(n).ne.canl_names_spec(29)
     &   .and.cnm(n).ne.canl_names_spec(31).and.cnm(n)
     &   .ne.canl_names_spec(36).and.cnm(n).ne.canl_names_spec(47)
     &   .and.cnm(n).ne.canl_names_spec(27).and.cnm(n)
     &   .ne.canl_names_spec(32).and.cnm(n).ne.'L67E '
     &   .and.cnm(n) .ne. 'L38E ') THEN
         IF (runmode.eq.'SIMUL'.or.cnm(n).ne.canl_names_spec(31)) THEN
           IF ((qu(n)-poutd).gt.tstor) THEN
             DO 40 i=1,nodcr(n)
               pond(nodec(i))=pond(nodec(i))+(qu(n)-poutd-tstor)/(gdar*
     &          nodcr(n)*(1.0-total_small_afact(nodec(i))-carfact
     &          (nodec(i))))
               rmonthly_ovlflo_to_canal(nodec(i))=
     &          rmonthly_ovlflo_to_canal(nodec(i))-(qu(n)-poutd-tstor)/
     &          (nodcr(n)*43560.)
   40        CONTINUE
             qud=qu(n)-poutd-tstor
             tovlf=qu(n)-poutd-tstor
```

```
              qu(n)=poutd+tstor
              if (ivary_slope_daily_opt(n).eq.1 .and. lcnb(n) .ge.1
     &          .and. lcnb(n) .le. 5) then
                call match(cnm(n),1,5,canal_name_dvslope,ncnl_dly_slope
     &           ,5,imatch_s)
                dischg_for_slope_kn(imatch_s,1,1) = dischg_for_slope_kn
     &            (imatch_s,1,1)- qud/86400.
              endif
            ELSE
              qud=0.0
              tovlf=0.0
            END IF
          ELSE
            qud=0.0
            tovlf=0.0
          END IF
        ELSE
          qud=0.0
          tovlf=0.0
        END IF
c
c   discontinue flood control releases thru oulet structure(s)
c   only when stages at all outlet structures are below respective flood
c   control triggers
c
        igrav_true_b = 0
        do istruc = 1,nstrctr(n)
          DO k = 1,no_match_loc(istruc)
            istrucm = imatch_struc_no(istruc,k)
            IF (ncoeff(n,istrucm).eq.1) THEN
              imgte=1
            ELSE
              imgte=iseas
            END IF
            if (lgrav(n,istrucm,1).and.irating(n,istrucm,1,imgte)
     $      .eq. 1 .and. ipurpse(n,istrucm,1).ne.1) igrav_true_b
     $      = igrav_true_b + 1
            if (nsubstr(n,istrucm) .gt. 1) then
              do isb  = 2,nsubstr(n,istrucm)
                if (.not. lgrav(n,istrucm,1).and.irating(n,istrucm,1
     $          ,imgte).eq. 1 .and. ipurpse(n,istrucm,1).ne.1) then
                 if (lgrav(n,istrucm,isb) .and. irating(n,istrucm,isb
     $            ,imgt) .eq. 1 .and. ipurpse(n,istrucm,isb)
     $            .ne. 1) then
                   igrav_true_b = igrav_true_b + 1
                   go to 5555
                 endif
                endif
              enddo
            endif
 5555       continue
          enddo
        enddo
c
        do istruc = 1,nstrctr(n)
           if (imatch_struc_loc(n,istruc) .eq. 0) then
            IF (ncoeff(n,istruc).eq.1) THEN
```

```
              imgte=1
            ELSE
              imgte=iseas
            END IF
            if (lgrav(n,istruc,1).and.irating(n,istruc,1,imgte)
     $      .eq. 1 .and. ipurpse(n,istruc,1).ne.1) igrav_true_b
     $      = igrav_true_b + 1
            if (nsubstr(n,istruc) .gt. 1) then
              do isb  = 2,nsubstr(n,istruc)
                if (.not. lgrav(n,istruc,1).and.irating(n,istruc,1
     $          ,imgte).eq. 1 .and. ipurpse(n,istruc,1).ne.1) then
                 if (lgrav(n,istruc,isb) .and. irating(n,istruc,isb
     $            ,imgt) .eq. 1 .and. ipurpse(n,istruc,isb)
     $            .ne. 1) then
                   igrav_true_b = igrav_true_b + 1
                   go to 5557
                 endif
                endif
              enddo
            endif
 5557       continue
          endif
        enddo
c       if (cnm(n).eq.'C4  ') print *,igrav_true_b,nstrctr(n),
c    $    n_gated_grav_outlets_fc(n),(lgrav(n,istruc,1),istruc=1
c    $    ,nstrctr(n))
c
c  pump outlets
c
        ipump_true_b = 0
        do istruc = 1,nstrctr(n)
          DO k = 1,no_match_loc(istruc)
            istrucm = imatch_struc_no(istruc,k)
            IF (ncoeff(n,istrucm).eq.1) THEN
              imgte=1
            ELSE
              imgte=iseas
            END IF
            if (lpump(n,istrucm,1).and.ipurpse(n,istrucm,1).ne.1)
     $       ipump_true_b = ipump_true_b + 1
            if (nsubstr(n,istrucm) .gt. 1) then
              do isb  = 2,nsubstr(n,istrucm)
                 if (.not. lpump(n,istrucm,1).and.ipurpse(n,istrucm,1)
     $           .ne.1) then
                   if (lpump(n,istrucm,isb).and.ipurpse(n,istrucm,isb)
     $             .ne.1) then
                     ipump_true_b = ipump_true_b + 1
                     GO TO 5556
                   endif
                 endif
              enddo
            endif
 5556       continue
          enddo
        enddo
c
        do istruc = 1,nstrctr(n)
```

```
            if (imatch_struc_loc(n,istruc) .eq. 0) then
              IF (ncoeff(n,istruc).eq.1) THEN
                imgte=1
              ELSE
                imgte=iseas
              END IF
              if (lpump(n,istruc,1).and.ipurpse(n,istruc,1).ne.1)
     $        ipump_true_b = ipump_true_b + 1
              if (nsubstr(n,istruc) .gt. 1) then
                do isb  = 2,nsubstr(n,istruc)
                  if (.not. lpump(n,istruc,1).and.ipurpse(n,istruc,1)
     $            .ne.1) then
                    if (lpump(n,istruc,isb).and.ipurpse(n,istruc,isb)
     $              .ne.1) then
                      ipump_true_b = ipump_true_b + 1
                      GO TO 5558
                    endif
                  endif
                enddo
              endif
 5558         continue
            endif
        enddo
!-----------------
!  DO CANAL CALCULATIONS NSTEP TIMES EACH DAY
!-----------------
        errprev=1
        iter=0
        if (rmax_dschg_coef(n) .gt.10000.) then
           fincr=0.25
        else
           fincr=1.0
        endif
        fac=fincr
        conv=0.
        hdrop_prev = HDC(N,iseas)
        idp=0
        jdp=0
        stage_prev=chdep(n)
        IF (ivary_slope_daily_opt(n) .eq. 1) then
           call match(cnm(n),1,5,canal_name_dvslope,
     &            ncnl_dly_slope,5,imatch_slope)
           ncdhd = imatch_slope
           if (igrid_loc_upslp(ncdhd) .gt. 0 .and. pond
     &      (igrid_loc_dnslp(ncdhd)) .ge. 0.50) then
              DPTHGW = (ELLS(igrid_loc_upslp(ncdhd))
     &        - H(igrid_loc_upslp(ncdhd))) * S(igrid_loc_upslp(ncdhd))
              IF (POND(igrid_loc_upslp(ncdhd)) .GT. DPTHGW) THEN
                 stgupslp = POND(igrid_loc_upslp(ncdhd)) - DPTHGW
     &           + ELLS(igrid_loc_upslp(ncdhd))
              ELSE
                 stgupslp = (POND(igrid_loc_upslp(ncdhd)) - DPTHGW)
     &           /s(igrid_loc_upslp(ncdhd))+ELLS(igrid_loc_upslp(ncdhd))
              ENDIF
              DPTHGW = (ELLS(igrid_loc_dnslp(ncdhd))
     &        - H(igrid_loc_dnslp(ncdhd))) * S(igrid_loc_dnslp(ncdhd))
              IF (POND(igrid_loc_dnslp(ncdhd)) .GT. DPTHGW) THEN
```

```
                 stgdnslp = POND(igrid_loc_dnslp(ncdhd)) - DPTHGW
     &           + ELLS(igrid_loc_dnslp(ncdhd))
              ELSE
                 stgdnslp = (POND(igrid_loc_dnslp(ncdhd)) - DPTHGW)
     &           /s(igrid_loc_dnslp(ncdhd))+ELLS(igrid_loc_dnslp(ncdhd))
              ENDIF
              HDC(n,iseas) = amax1(stgupslp - stgdnslp,0.0)
     &        *hdrop_fact(ncdhd)
              HDC(n,iseas)= min(HDC(n,iseas),hdrop_max(ncdhd))
c             print *,cnm(n),chdep(n),HDC(n,iseas),'  WCA SLOPE  '
c    &         ,stgupslp,pond(igrid_loc_upslp(ncdhd)),stgdnslp
c    &         ,pond(igrid_loc_dnslp(ncdhd))
           endif
        ENDIF
   50   dsum=0.
!-----------------
!  BEGIN ITERATION FOR EACH CANAL
!-----------------
        cout=0.0
        out_flow_east = out_flow_east0
        coutd=0.0
        ovlfl=0.0
        cgvol=0.0
        tdmnd=0.0
        nstideout=0
        tot_seep_inflw = 0.0
        IF (ivary_slope_daily_opt(n) .eq. 1) then
           do bb = 1,2
            do cc = 1,3
             dischg_for_slope(ncdhd,bb,cc) =
     &        dischg_for_slope_in(ncdhd,bb,cc)
            enddo
           enddo
        ENDIF
        DO kk = 1,nstrctr(n)
          IF (imatch_struc_loc(n,kk) .ne. 0) THEN
            i_counter_m(imatch_struc_loc(n,kk)) = 0
            i_counter_mx(imatch_struc_loc(n,kk)) = 0
          ENDIF
        ENDDO
!
! COMPUTE LVSEEP INTO L-30
!
        if (cnm(n) .eq. 'L30  ') then
         do ilv = 1,no_of_l30spnod
           if (icol_f_l30(ilv) .eq. 27 .and. irow_f_l30(ilv)
     &     .eq. 27) then
              stgdn_sp = (chdep(l30) + ((no_of_l30spnod-ilv)
     &         /(no_of_l30spnod-1))*hdc(l30,iseas) + chdep(isncre)
     &         + hdc(isncre,iseas))/2.0
           else
              stgdn_sp = chdep(l30) + ((no_of_l30spnod-ilv)
     &        /(no_of_l30spnod-1))*hdc(l30,iseas)
           endif
           hdiffsp = HWSEEP_F_L30(ilv) - stgdn_sp
           hdiffusp = HWSEEP_F_L30(ilv) - HDSEEP_F_L30(ilv)
           if (srate(ilv_f_l30(ilv),2) .lt. 0.0) THEN
```

```
            HD_DROP_N = amax1(hdiffusp,0.0)
         else
            HD_DROP_N = hdiffusp
         endif
         if (srate(ilv_f_l30(ilv),1) .lt. 0.0) THEN
            HD_DROP = amax1(hdiffsp,0.0)
         else
            HD_DROP = hdiffsp
         endif
         seep_f_l30 = SRATE(ilv_f_l30(ilv),1) * HD_DROP
     &    + SRATE(ilv_f_l30(ilv),2) * HD_DROP_N
     &    + SRATE(ilv_f_l30(ilv),3)
         seep_f_l30 = seep_f_l30 * srate_frac(ilv_f_l30(ilv))
         seep_f_l30 = amin1(seep_f_l30,rate_limit(ilv_f_l30(ilv)))
         seep_f_l30_1 = seep_f_l30 * dy /5280.
         seep_f_l30 = amax1(seep_f_l30_1,0.0)
c        if (conv.eq.1.)
c    &   print *,seep_f_l30,HD_DROP,HD_DROP_N,stgdn_sp,'  PPPP  '
c    &   ,HWSEEP_F_L30(ilv),HDSEEP_F_L30(ilv),icol_f_l30(ilv)
c    &   ,irow_f_l30(ilv)
         if (icol_f_l30(ilv) .eq. 27 .and. irow_f_l30(ilv)
     &   .eq. 27) THEN
           tot_seep_inflw = seep_f_l30*0.40 + tot_seep_inflw
           IF (ivary_slope_daily_opt(L30) .eq. 1) THEN
             call match(cnm(L30),1,5,canal_name_dvslope
     &       ,ncnl_dly_slope,5,imatch_slope)
             dischg_for_slope(imatch_slope,1,2)
     &       = dischg_for_slope(imatch_slope,1,2)
     &       + seep_f_l30 * 0.40
           ENDIF
         else
           tot_seep_inflw = seep_f_l30 + tot_seep_inflw
           call match(cnm(L30),1,5,canal_name_dvslope
     &     ,ncnl_dly_slope,5,imatch_slope)
           IF (ivary_slope_daily_opt(L30) .eq. 1) THEN
              IF (ilv .LE. mid_point_canl(L30)) THEN
                 dischg_for_slope(imatch_slope,1,2)=dischg_for_slope
     $            (imatch_slope,1,2) + seep_f_l30
              ELSE
                 dischg_for_slope(imatch_slope,1,3)=dischg_for_slope
     $            (imatch_slope,1,3) - seep_f_l30
              ENDIF
           ENDIF
         endif
         if (conv .eq. 1.) then
           icell_no_spl30 = icol_f_l30(ilv) - minx(irow_f_l30(ilv))
     &     + 1 + isum(irow_f_l30(ilv))
           rchg(icell_no_spl30) = rchg(icell_no_spl30) -
     &     amax1(seep_f_l30*86400./gdar - pond(icell_no_spl30),0.0)
           pond(icell_no_spl30) = pond(icell_no_spl30) - amin1(
     &     seep_f_l30*86400./gdar,pond(icell_no_spl30))
           IF (icol_f_l30(ilv) .eq. 27 .and. irow_f_l30(ilv)
     &     .eq. 27) THEN
              NODE2826 =  28 - MINX(26) + 1 + ISUM(26)
              IF(ELLS(NODE2826) .GT. 0.0) THEN
                 QU (ISNCRE) = QU (ISNCRE) + seep_f_l30 * 86400.
     &           * 0.60
```

```
                  rlvseep_to_l30 = rlvseep_to_l30
     &             + seep_f_l30
                  IF (ivary_slope_daily_opt(ISNCRE) .eq. 1) THEN
                    call match(cnm(ISNCRE),1,5,canal_name_dvslope
     &              ,ncnl_dly_slope,5,imatch_slope)
                    dischg_for_slope_kn(imatch_slope,1,1)
     &              = dischg_for_slope_kn(imatch_slope,1,1)
     &              + seep_f_l30 * 0.60
                  ENDIF
               ENDIF
            ELSE
              rlvseep_to_l30 = rlvseep_to_l30 + seep_f_l30
            ENDIF
            daily_seep(ipos_lv_l30(ilv)) = seep_f_l30*1.9835
            if (.not. day1) then
               seepmon(ipos_lv_l30(ilv)) = seepmon(ipos_lv_l30(ilv))
     &         + daily_seep(ipos_lv_l30(ilv))
            endif
          endif
        enddo
        total_inflw_l30 = qutemp + tot_seep_inflw*86400.
        qu(l30) = total_inflw_l30
      endif
!
!
      IF (nint(crmin(n,iseas)).ne.-9 .and. nint(crmin(n,iseas))
     &  .ne.-10) THEN
        av_cstor=(chdep(n)-stg_min_ws_dschg(n))*carea+cgseep(n,
     &   stg_min_ws_dschg(n),0,0.)+tovflc(n,stg_min_ws_dschg(n),0,0.,
     &   iter,1)
        IF (ws_status_cnl(n).ne.'FLWTH') THEN
          cstor_def=(stg_min_ws_dschg(n)-chdep(n))*carea-cgseep(n,
     &     stg_min_ws_dschg(n),0,0.)-tovflc(n,stg_min_ws_dschg(n),0,
     &     0.,iter,1)
          cstor_def=max(cstor_def,0.0)
        ELSE
          cstor_def=0.0
        END IF
        avol(n)=max(av_cstor,0.0)+qu(n)-max(cstor_def,0.0)
        avol(n)=max(avol(n)/86400.0,0.0)
      END IF
!
!------------------
!  BEGIN ITERATION ALONG NODES OF A SINGLE CANAL
!------------------
!    CALCULATE OVERLAND FLOW FOR REACH
!-----------------------------
c     if(cnm(n).eq.'C13 ')print *,iter,conv,HDC(N,iseas),'  HDC'
c    $ , chdep(n)
      ovlfl=tovflc(n,chdep(n),1,conv,iter,1)
!
!-------------------------------
!    CALCULATE TOTAL SEEPAGE
!-----------------------------
!
      cgvol=cgseep(n,chdep(n),1,conv)
!
```

```
!-------------------
!     CALCULATE INJECTION INTO ASR WELLS ,IF ANY
!-------------------
        cout_asr=0.0
        IF (nasr_from_cnl(n).gt.0) THEN
          IF ((ndays_prestorm_drawdown .eq. 0 .OR. nstorm_day
     &    (ndays_sim_sd).eq. 0 .OR. icanal_drawdwn .eq. 0 )
     &    .and. ivary_slope_daily_opt(n).eq.1) THEN
             IF (canl_rf_avg(n) .gt. 5.50/14.) then
                asr_inj_trig_drwdn = rf_drawdwn_dph(n,2,iseas)
             elseif (canl_rf_avg(n) .ge. 3.80/14.) then
                asr_inj_trig_drwdn = rf_drawdwn_dph(n,1,iseas)
             else
                asr_inj_trig_drwdn = 0.0
             endif
          ELSE
            asr_inj_trig_drwdn = 0.0
          ENDIF
          DO nasr=1,nasr_from_cnl(n)
            trig_inj_to_asr_1 = cnl_trig_inj_to_asr(n,nasr)
     &      - asr_inj_trig_drwdn
            IF (chdep(n).gt.trig_inj_to_asr_1+0.2) THEN
              cout_to_asr=asr_incap(asrnum(1,n,nasr,1))
            ELSE IF (chdep(n).gt.trig_inj_to_asr_1) THEN
              cout_to_asr=5.0*(chdep(n)-trig_inj_to_asr_1)*
     &         asr_incap(asrnum(1,n,nasr,1))
            ELSE
              cout_to_asr=0.0
            END IF
            IF (conv.eq.1.) THEN
              vol_in=cout_to_asr*3.089
              CALL asr (asrnum(1,n,nasr,1), 1, vol_in, vol_not_inj, 0,
     &         tot_cnl_dmd_asr)
            END IF
            cout_asr=cout_asr+cout_to_asr*1.547*86400.
          END DO
          IF (conv.eq.1.) THEN
            IF (cnm(n).eq.canl_names_spec(12)) THEN
              vol_inj_to_c51asr=cout_asr/86400.
            ELSE IF (cnm(n).eq.canl_names_spec(22)) THEN
              vol_inj_to_hlsbasr=cout_asr/86400.
            ELSE IF (cnm(n).eq.canl_names_spec(40)) THEN
              vol_inj_to_nnrcasr=cout_asr/86400.
            ELSE
              vol_inj_to_l31asr=vol_inj_to_l31asr+cout_asr/86400.
            END IF
          END IF
c           if (conv .eq. 1.)
c    $      print *,cnm(n),cout_to_asr*1.547,chdep(n),rmnstgs(n,1,iseas)
c    $       ,trig_inj_to_asr_1,'  CNLASR '
        END IF
        IF (ivary_slope_daily_opt(n) .eq. 1) then
          do bb = 1,2
            do cc = 1,3
              dischg_for_slope_max(ncdhd,bb,cc) =
     &        dischg_for_slope(ncdhd,bb,cc)
            enddo
```

```
          enddo
        ENDIF
!-----------------
!     CALCULATE OUTFLOW (COUT) AT DOWNSTREAM STRUCTURE LOCATED AT KX,KY
!-----------------
        IF (into(n,1).ge.0) THEN
          toutprev=0.0
          itideout=0
          DO 70 jj=1,nout(n)
            in=into(n,jj)
!
!  take into account the slope of canal
!
            if (nodcr(n).gt.1)then
              stgdiff_canal_ups=float(nodcr(n)-nodepos_wu(n,jj))/
     &         float(nodcr(n)-1)*hdc(n,iseas)
            else
              stgdiff_canal_ups=0.0
            endif
            hw=chdep(n)+stgdiff_canal_ups
!
            IF (in.eq.999.or.in.le.0.or.in.eq.888) THEN
!CCC FOR CANAL DISCHARGING INTO NODE
              IF (in.ne.0) THEN
                crelev=crel(n,jj)+stgdiff_canal_ups
                IF (in .eq. 999) crelev = crel(n,jj)
              ELSE
                itideout=itideout+1
                twdph=get_tide(ycnlnode(n,itideout),jday,
     &           proj_sea_level_rise)
                twdph = 1.025*(twdph +2.0) - 2.0
                crelev=max(crel(n,jj)+stgdiff_canal_ups,twdph)
              END IF
              IF (in .ne. 999) THEN
                 IF (hw.gt.crelev) THEN
                   IF (weir_name(n,jj).ne.'G93  ') THEN
                     cout=strcf(n,jj)*86400.*gwdth(n,jj)*(hw-crelev)
     &               *sqrt(hw-crelev)
                   ELSE
                     cout=strcf(n,jj)*86400.*gwdth(n,jj)
     &               *sqrt(hw-crelev)
                   ENDIF
                 ELSE
                   cout=0.0
                 END IF
              ELSE
                lstnode = NODE_WD(n,jj)
                deptgw = ells(lstnode) - h(lstnode)
                if (pond(lstnode) .gt. deptgw) then
                   tw = pond(lstnode) - deptgw + ells(lstnode)
                else
                   tw = (pond(lstnode) - deptgw)/s(lstnode)
     &              + ells(lstnode)
                endif
                IF (tw .le. crelev) THEN
                  IF (hw.gt.crelev) THEN
                    vol_lim_w = (hw - ells(lstnode))* gdar * 0.80
```

```
                    cout = strcf(n,jj)*86400.*gwdth(n,jj)*(hw-crelev)
     &              *sqrt(hw-crelev)
                    cout = amin1(cout,vol_lim_w)
                  ELSE
                    cout = 0.0
                  ENDIF
                ELSE
                  IF (hw .lt. crelev) THEN
                     sgn = -1.0
                     vol_lim_w = amax1(pond(lstnode)-deten(lstnode),0.0)
     &               *gdar*(1.0-carfact(lstnode))
                     cout = sgn*strcf(n,jj)*86400.*gwdth(n,jj)*(tw
     &               -crelev)*sqrt(tw-crelev)
                     cout = sgn*amin1(sgn*cout,vol_lim_w)
                  ELSE
                     IF (hw .gt. tw) THEN
                        submerg_fact = (tw - crelev)/(hw - crelev)
                        rel_crst_length = (hw - crelev)/(hw - crelev
     &                  + 5.0)
                        rmod_limit = 0.85 - 0.50*rel_crst_length
                        reduc_fact = (submerg_fact -rmod_limit)
     &                  /(1.0 -rmod_limit)
                        reduc_fact = amax1(reduc_fact,0.0)
                        reduc_fact = (1.0 - reduc_fact) ** 0.25
                        vol_lim_w = (hw - tw) * gdar * 0.80
                        cout = strcf(n,jj)*86400.*gwdth(n,jj)*(hw
     &                  -crelev)*sqrt(hw-crelev)*reduc_fact
                        cout = amin1(cout,vol_lim_w)
                     ELSE
                        sgn = -1.0
                        submerg_fact = (hw - crelev)/(tw - crelev)
                        rel_crst_length = (tw - crelev)/(tw - crelev
     &                  + 5.0)
                        rmod_limit = 0.85 - 0.50*rel_crst_length
                        reduc_fact = (submerg_fact -rmod_limit)
     &                  /(1.0 -rmod_limit)
                        reduc_fact = amax1(reduc_fact,0.0)
                        reduc_fact = (1.0 - reduc_fact) ** 0.25
                        cout = sgn*strcf(n,jj)*86400.*gwdth(n,jj)
     &                  *(tw-crelev)*sqrt(tw-crelev)*reduc_fact
                        vol_lim_w = max(ells(lstnode)+pond(lstnode)
     &                  -max(hw,ells(lstnode)+deten(lstnode)),0.0)
     &                   *gdar*0.8
                        cout = sgn*amin1(sgn*cout,vol_lim_w)
                     ENDIF
                    ENDIF
                  ENDIF
                ENDIF
c               IF (conv.eq.1..and.in.eq.999) THEN
c                    pond(lstnode)=cout/(gdar*(1.0-carfact(lstnode)))
c    &               +pond(lstnode)
c               END IF
                IF (conv.eq.1..and.in.eq.999) THEN
                  IF (cnm(n).eq.'C304 ') c304_out_to_marsh = cout/86400.
                ENDIF
c
c  compute discharge for canals with daily varying slope
```

```
c
              IF (ivary_slope_daily_opt(n) .eq. 1) then
                sgn = 1
                IF (iwuppflw_for_slope(n,jj) .eq. 2) then
                   sgn = -1
                ENDIF
                 dischg_for_slope(ncdhd,2,iwuppflw_for_slope(n,jj))
     &           = dischg_for_slope(ncdhd,2,iwuppflw_for_slope(n,jj))
     &           + sgn*cout/86400.
              ENDIF
            ELSE
              IF (iter.eq.0) rnflwin(n,jj)=qu(in)
              crelev=crel(n,jj)+stgdiff_canal_ups
              IF (hw.gt.crelev) THEN
                cout=strcf(n,jj)*86400.*gwdth(n,jj)
     &           *sqrt(hw-crelev)
              ELSE
                cout=0.0
              END IF
              coutlim=0.0
              IF (into(in,1).ge.0) THEN
                itide_down=0
                DO nd=1,nout(in)
                  icanal_down=into(in,nd)
                  IF (icanal_down.eq.0) THEN
                    itide_down=itide_down+1
                    twdph_down=get_tide(ycnlnode(in,itide_down),jday,
     &               proj_sea_level_rise)
                    twdph_down=1.025*(twdph_down + 2.0) -2.0
                    crelev_down=amax1(crel(in,nd),twdph_down)
                  ELSE
                    crelev_down=crel(in,nd)
                  END IF
                  IF (hw.gt.crelev_down) THEN
                    IF (into(in,1).eq.0) THEN
                     coutdn=strcf(in,nd)*gwdth(in,nd)*(hw-crelev_down)*
     &               sqrt(hw-crelev_down)
                    ELSE
                     coutdn=strcf(in,nd)*gwdth(in,nd)*sqrt
     &               (hw-crelev_down)
                    ENDIF
                  ELSE
                    coutdn=0.0
                  END IF
                  coutlim=coutlim+coutdn
                END DO
              END IF
              tot_pump=0.0
              DO nd=1,nstrctr(in)
                coutlim=strout(hw,n,in,nd,0,crmin,volwtr,prevqu,rain,
     &          iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)+coutlim
              END DO
              if (nodcr(in) .gt.1) then
                stgdiff_canal_down=float(nodcr(in)-nodepos_wd(n,jj))/
     &          float(nodcr(in)-1)*hdc(in,iseas)
              else
                stgdiff_canal_down=0.0
```

```
              endif
              tw=chdep(in)+stgdiff_canal_down
!
              avvoldn=(hw-tw)*cl(in)*width(in)-cgseep(in,hw-
     &         stgdiff_canal_down,0,0.)-tovflc(in,hw-stgdiff_canal_down,
     &         0,0.,iter,2)-rnflwin(n,jj)+coutlim*86400.
              avvoldn=amax1(avvoldn,0.0)
              IF (jj.eq.1.or.(jj.gt.1.and.in.ne.into(n,jj-1))) THEN
                avvol_ds=avvoldn
              ELSE
                avvol_ds=avvoldn-toutprev
                avvol_ds=amax1(avvol_ds,0.0)
              END IF
              couttmp=amin1(cout,avvol_ds)
              IF ((jj.gt.1.and.in.eq.into(n,jj-1))
     &         .or.(jj.eq.1.and.into(n,jj+1).eq.in)) THEN
                toutprev=couttmp+toutprev
              END IF
              cout=couttmp
            END IF
            IF (cnm(n).eq.canl_names_spec(35)) cout=amin1(cout,250.0*
     &       86400.)
!
            coutd=coutd+cout
c
c  compute discharge for calculation of daily varying slope
c
              IF (ivary_slope_daily_opt(n) .eq. 1) then
                 sgn = 1
                 IF (iwuppflw_for_slope(n,jj) .eq. 2) then
                    sgn = -1
                 ENDIF
                 dischg_for_slope(ncdhd,2,iwuppflw_for_slope(n,jj))
     &           = dischg_for_slope(ncdhd,2,iwuppflw_for_slope(n,jj))
     &           + sgn*cout/86400.
              ENDIF
!              if (ifyr + iyear-1 .eq. 1981 .and.month.ge.6)then
!               if (conv.eq.1..and.n.eq.72)
!    $          print *,COUTD/86400.,COUT/86400.
!              endif
!
!-------
            IF (conv.eq.1) THEN
              IF (into(n,jj).eq.ic51w) drainc51w=drainc51w+cout
              IF (into(n,jj).eq.ic11w) drainc11w=drainc11w+cout
              IF (into(n,jj).eq.ic51) drainc51=drainc51+cout
              IF (into(n,jj).eq.lhl) drainhil=drainhil+cout
c
c    compute discharge for calculation of daily varying slope of
c    downstream canal
c
              IF (in .le. nch .and. in .ne. 0) then
                IF (ivary_slope_daily_opt(in) .eq. 1) then
                  call match(cnm(in),1,5,canal_name_dvslope,
     &            ncnl_dly_slope,5,imatch_slope)
                 sgn = 1
                 IF (iwdnpflw_for_slope(n,jj) .eq. 3) then
```

```
                  sgn = -1
                ENDIF
                dischg_for_slope_in(imatch_slope,1,iwdnpflw_for_slope
     &          (n,jj)) = dischg_for_slope_in(imatch_slope,1,
     &          iwdnpflw_for_slope(n,jj)) + sgn*cout/86400.
              ENDIF
            ENDIF
c
c           jstr1=jstr1+1
c           strflw(jstr1,idss)=cout/86400.
            weir_struc_f_dss(jj) = cout/86400.
            IF (into(n,jj).eq.ic51w.and.cnm(n).eq.canl_names_spec(46))
     &        THEN
              tot_avl_l8_bpit_res=tot_avl_l8_bpit_res+cout/86400.
              tot_avl_l8_bpit_res_lwd1=cout/86400.
            END IF
            IF (cnm(n).eq.canl_names_spec(49)) THEN
              outflow_from_us27s=cout/86400.
            END IF
            IF (cnm(n).eq.canl_names_spec(50)) THEN
              outflow_from_us27n=cout/86400.
            END IF
          END IF
!-------
          IF (in.eq.999.or.in.le.0.or.in.eq.888) GO TO 70
          IF (conv.eq.1.) THEN
            qu(in)=qu(in)+cout
            upstream_inflow(in)=upstream_inflow(in)+amax1(cout/86400.-
     &       dqu(in),0.0)*86400.
            IF (iserv_area_loc(in).gt.0) THEN
              IF (in.gt.n) THEN
                rloc_excess_for_ws(iserv_area_loc(in))=
     &           rloc_excess_for_ws(iserv_area_loc(in))+amin1(cout/
     &           86400.,dqu(in))*86400.
              ELSE
                rloc_excss_to_canl_next_step(in)=
     &           rloc_excss_to_canl_next_step(in)+cout
              END IF
            END IF
            cout_for_ws=amin1(cout/86400.,dqu(in))
            dqu(in)=dqu(in)-cout_for_ws
            icnl_dns=in
   60       icnl_dns=iup_canal_ws(icnl_dns)
            IF (icnl_dns.le.nch) THEN
              dqu(icnl_dns)=dqu(icnl_dns)-cout_for_ws
              dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
              GO TO 60
            END IF
          END IF
   70   CONTINUE
        DO jjk = 1,nout(n)
          lstnode1 = NODE_WD(n,jjk)
          IF (conv .eq.1. .and. into(n,jjk) .eq. 999) THEN
            pond(lstnode1)=weir_struc_f_dss(jjk)*86400./(gdar*(1.0-
     &      carfact(lstnode1)))+pond(lstnode1)
          ENDIF
        ENDDO
```

```
        END IF
!
        IF (nstrctr(n).gt.0) THEN
          DO 80 in=1,nstrctr(n)
!
            IF (flnm(istr(n,in)).eq.'S332  '.and.opt_flow_to_ts.eq.'TSMI
     &NDL') THEN
              adddmnd=vol332(month)
            ELSE IF (flnm(istr(n,in)).eq.'S332  '
     &       .and.opt_flow_to_ts.ne.'TSMINDL'.and.opt_flow_to_ts.ne.'TSR
     &FPLN') THEN
              adddmnd=vol332do(month)
            ELSE IF (flnm(istr(n,in)).eq.'LXSLWS'.and.lwslo(iloxah))
     &       THEN
              adddmnd=discoef(n,in,1,1)
            ELSE IF (flnm(istr(n,in)).eq.'G92TRV') THEN
              adddmnd=strdcap(n,in,1,1)
            ELSE
              adddmnd=0.0
            END IF
            DO i=1,no_ds_cnl_ws(n,in)
!
              IF (ipurpse(n,in,1).ne.2) THEN
                IF (flnm(istr(n,in)).eq.'S334  ') THEN
                  ds_wsneed=ws_need_l31nc_orig
                ELSE
                  ds_wsneed=tdqu_dn(idncnl_ws(n,in,i))
                END IF
                tdmnd=tdmnd+ds_wsneed
              END IF
            END DO
            tdmnd=tdmnd+adddmnd
   80     CONTINUE
          tdmnd_orig=tdmnd
c         IF (runmode.eq.'CALIB') THEN
          IF (cnm(n).eq.canl_names_spec(31)) THEN
           call match('S334  ',1,6,struc_name_meas,nflpts,6,imatch_s334)
           call match('S336  ',1,6,struc_name_meas,nflpts,6,imatch_s336)
           call match('S338  ',1,6,struc_name_meas,nflpts,6,imatch_s338)
           IF (ifix(kflo(ifls334)).gt.0.or.tdmnd.gt.0.01) THEN
              IF (imatch_s334 .ne. 0) THEN
                tdmnd=tdmnd-kflo(ifls334)
              ENDIF
              IF (imatch_s336 .ne. 0) THEN
                tdmnd=tdmnd+kflo(ifls336)
              ENDIF
              IF (imatch_s338 .ne. 0) THEN
                tdmnd=tdmnd+kflo(ifls338)
              ENDIF
              tdmnd=amax1(tdmnd,0.0)
           END IF
          END IF
c         END IF
          IF (nint(crmin(n,iseas)).ne.-10 .and. nint(crmin(n,iseas))
     &     .ne.-9) THEN
           tempvol=avol(n)
          ELSE
```

```
          IF (cnm(n).eq.canl_names_spec(47)) THEN
            tempvol=s333_ws
          ELSE
            tempvol=qu(n)/86400.
          END IF
          IF (iter.eq.0) THEN
            IF (ws_status_cnl(n).eq.'FLWTH') THEN
              IF (iserv_area_loc(n).gt.0) THEN
                rloc_seep_for_ws(iserv_area_loc(n))=
     &           rloc_seep_for_ws(iserv_area_loc(n))+amin1(amax1(qu(n)
     &           -upstream_inflow(n),0.0),amax1(dqu(n)*86400,0.0))
              END IF
            END IF
          END IF
        END IF
        tdmnd_ds=tdmnd
!
        IF (opt_fc_canal(n).eq.'FLWTH') THEN
          IF (cnm(n).eq.canl_names_spec(47)) THEN
            avail_fc_vol=s333_reg_lec
          ELSE
            avail_fc_vol=qu(n)/86400.
          END IF
        ELSE
          avail_fc_vol=99999.
        END IF
        DO ind=1,nstrctr(n)
           DO ires = 1,NRES(n,ind)
              TIVOLWTR(IRSRCVR(N,IND,IRES))
     &        = VOLWTR(IRSRCVR(N,IND,IRES))
           ENDDO
        ENDDO
        total_flow_s12=0.0
        DO 90 ind=1,nstrctr(n)
          IF (nprty(n,ind).ne.0) THEN
            ind1=nprty(n,ind)
          ELSE
            ind1=ind
          END IF
          IDNCNL(N,IND1) = IDNCNL2(N,IND1)
          IRCEIVE(N,IND1) = IRCEIVE2(N,IND1)
          if (nint(s337fc) .gt. 0 .or. nint(s333_reg_lec)
     &     .gt. 0) then
            no = 1
            do no1 = 1,ndual_canal_ops(no)
              if (ind1 .eq. istruc_no_dualops(no1,no)
     &         .and. n .eq. icanal_no_dualops(no1,no)) then
                IF (ndual_col_dest(no1,no) .lt. 0) THEN
                  IDNCNL(N,IND1) = ndual_canl_dest_indx(no1,no)
                  IRCEIVE(N,IND1) = 1
                ELSEIF (IRCEIVE(N,IND1) .EQ. 1) THEN
                  NODRCV(N,IND1,1) = igrid_loc_dest_dualops(no1,no)
                  IRCEIVE(N,IND1) = 2
                  ncells_rec(n,ind1) = 1
                ENDIF
              endif
            enddo
```

```
        endif
        if (nint(s333_reg_lec) .gt. 0) then
           no = 2
           do no1 = 1,ndual_canal_ops(no)
             if (ind1 .eq. istruc_no_dualops(no1,no)
&            .and. n .eq. icanal_no_dualops(no1,no)) then
               IF (ndual_col_dest(no1,no) .lt. 0) THEN
                 IDNCNL(N,IND1) = ndual_canl_dest_indx(no1,no)
                 IRCEIVE(N,IND1) = 1
               ELSEIF (IRCEIVE(N,IND1) .EQ. 1) THEN
                 NODRCV(N,IND1,1) = igrid_loc_dest_dualops(no1,no)
                 IRCEIVE(N,IND1) = 2
                 ncells_rec(n,ind1) = 1
               ENDIF
             endif
           enddo
        endif
          if (irceive(n,ind1) .eq. 1) then
           if (n .eq. intcnl(idn_wca_indx(is8bsn-6),
&          iconvey_canal_no(is150bws-6))) then
             if (iconvey_canal_no(is150bws-6) .ne.
&            iconvey_canal_no(is8bsn-6)) then
               do mm = 1,nclns(idn_wca_indx(is8bsn-6),
&              iconvey_canal_no(is150bws-6))
                 if (IDNCNL(N,IND1) .eq. isacnl(idn_wca_indx(is8bsn
&                -6),iconvey_canal_no(is150bws-6),mm)) then
                   if (lwslo_floor(iconvey_canal_no(is150bws-6)))
&                  then
                     TEMPVOL = TINFLOW(idn_wca_indx(is150bws-6),
&                    iconvey_canal_no(is150bws-6))
&                    - outflow_from_3a(iconvey_canal_no(is150bws-6))
                   endif
                 endif
               enddo
             else
               floor = floor_elev(idn_wca_indx(is8bsn-6),
&                iconvey_canal_no(is150bws-6))
               do mm = 1,nclns(idn_wca_indx(is8bsn-6),
&              iconvey_canal_no(is150bws-6))
                 if (IDNCNL(N,IND1) .eq. isacnl(idn_wca_indx(is8bsn
&                -6),iconvey_canal_no(is150bws-6),mm)) then
                    iconvey_c = iconvey_canal_no(is150bws-6)
                    CNLSTOR=(CHDEP(INTCNL(IWCA3A,iconvey_c))-floor)
&                   *CL(INTCNL(IWCA3A,iconvey_c))*WIDTH(INTCNL
&                   (IWCA3A,iconvey_c))
                    AVAL_VOL = CNLSTOR+CGSEEP(INTCNL(IWCA3A
&                   ,iconvey_c),floor,0,0.)+TOVFLC(INTCNL(IWCA3A
&                   ,iconvey_c),floor,0,0.,iter,1)
                    AVVOL_TOT = AMAX1(AVAL_VOL,0.0)/86400. + TINFLOW
&                   (IWCA3A,iconvey_c)
                    AVVOL_TOT = AMAX1(AVVOL_TOT,0.0)
                    TEMPVOL = amax1(AVVOL_TOT - outflow_from_3a
&                   (iconvey_canal_no(is150bws-6)),0.0)
                 endif
               enddo
             endif
           endif
```

```
                endif
!
              CALL canl_dep_struc_param_setup(n,ind1,idncl
     $        ,capacws,capacfl,wsdmnd,imgt,LOGDISG,tdqu_dn
     $        ,ndsglim,nodedlim,fl_trig_constrt,tdmnd_orig
     $        ,hws,tws,gravity_flow_struc,cout,struc_name_meas
     $        ,LPMPS309,STGDIFF_CANAL_TW,itideout,nstideout,jday)
              CALL canl_dep_struc_capac_setup(n,ind1,idncl,conv
     $        ,crmin,iter,rain,prevqu,volwtr,tempf_volwtr,jday,TDQU_DN
     $        ,tot_flw_to_c9r_frm_nlkblt,rnlkblt_to_s29
     $        ,rlvseep_to_cnl,is334_open_flg,s334_fc,capacfl
     $        ,capacws,tempf_weircff,wsdmnd,imgt,hws,tws,STGDIFF_CANAL_TW
     $        ,gravity_flow_struc,flow_to_c9r_frm_c11,tot_grav_cap
     $        ,canl_rf_avg,s337fc)
              CALL gen_canl_dep_struc_flw(n,ind1,idncl,crmin
     $        ,rain,TIVOLWTR,capacws,capacfl,prevqu,VOLWTR,TDQU_DN
     $        ,outflow_from_3a,outflow_from_3a_temp,temp_volwtr
     $        ,tempf_volwtr,temp_weircff,tempf_weircff,tempvol
     $        ,total_small_afact,rloc_excess_for_ws,s334_fc
     $        ,rloc_excss_to_canl_next_step,node_wd,LOGDISG
     $        ,is334_open_flg,conv,jday,iter,tot_avl_l8_bpit_res_lwd1
     $        ,flow_to_ca1_frm_acmebsn,tdmnd,finstor,finstor_res
     $        ,flow_to_c9r_frm_c9,flow_to_lbn_frm_c11,wratio
     $        ,tot_flw_to_c9r_frm_nlkblt,avail_fc_vol
     $        ,flow_to_cpbres_frm_e1,dischg_to_bay_s29_frm_c11r
     $        ,frac_avail_cap_s334,cout,wsdmnd,imgt,hws,tws
     $        ,gravity_flow_struc,flow_to_c9r_frm_c11,tot_grav_cap
     $        ,s336_to_biscbay,flow_to_acmres_frm_basin,STGDIFF_CANAL_TW
     $        ,qinfldn,coutlim,avvol_dns334,add_s333_fc_s334
     $        ,tot_avl_l8_bpit_res_m1q,s337fc,outf_s9a)
              CALL spec_canl_dep_struc_flw(n,ind1,idncl,iter,LOGDISG
     $        ,conv,rain,TIVOLWTR,capacws,capacfl,volwtr,prevqu
     $        ,tempf_volwtr,tempf_weircff,rloc_excess_for_ws,tdqu_dn
     $        ,rloc_excss_to_canl_next_step,canl_rf_avg,s333s334rg_offset
     $        ,rlist_lvname,convey_canal_names,struc_name_meas,s334_fc
     $        ,iroute_to_wca3a_frm_holy,iseep_lim_s356_opt,tempvol
     $        ,frac_avail_cap_s333,is334_open_flg,s356_to_l29
     $        ,s356_to_cell,s337fc,cout,FINSTOR,finstor_res,jday,tinflow
     $        ,outflow_from_3a,tdmnd_ds,ws_to_l8_via_s352,wratio,imgt
     $        ,nconvey_total,pctmet,RMFLWS12,RMFLWS333,stagelo_beg
     $        ,outflow_from_3a_temp,s12_reg_total,s12_env_total
     $        ,flow_to_lbn_frm_c11,s333_reg_lec,s355prevout,prev_TOTF
     $        ,s333_reg_lec_abve_sch,s333_reg_lec_bel_sch,s333ftargprev
     $        ,total_flow_to_mcnl,ihly_out_count,stgref,total_flow_s12
     $        ,rfls155a,prev_total_flow_s12,hws,tws,tdmnd,crmin
     $        ,s336_to_biscbay,flow_to_acmres_frm_basin,LPMPS309
     $        ,STGDIFF_CANAL_TW,qinfldn,coutlim,avvol_dns334
     $        ,add_s333_fc_s334,stage_g3273,total_small_afact
     $        ,lflwth, l334rg, lwsdel, add_dmndagl8,prev_voldn_s334
     $        ,outf_s9a,out_flow_east,out_flow_east0
     $        ,iroute_acme_fc_to_sta1e_opt,frac_s319_sta1ew
     $        ,frac_s319_sta1ee,iroute_l8rnf_to_sta_opt)
              coutd=coutd+cout*86400.
c             if (cnm(n).eq.'S9UP ') write(445,*)cout,flnm(istr(n,ind))
c    $         ,'  S9UP OUT'
c
c  compute discharge for canals with daily varying slope
```

```
c
             IF (ivary_slope_daily_opt(n) .eq. 1) THEN
                sgn = 1
                IF (isuppflw_for_slope(n,ind) .eq. 2 .or.
     &           (isuppflw_for_slope(n,ind) .eq. 1 .and.
     &           isupp_at_ups_end(n,ind) .eq. 1)) THEN
                  sgn = -1
                ENDIF
                dischg_for_slope(ncdhd,2,isuppflw_for_slope(n,ind))
     &          = dischg_for_slope(ncdhd,2,isuppflw_for_slope(n,ind))
     &          + sgn*cout
             ENDIF
!
            IF (conv.eq.1) THEN
c
c    compute discharge for calculation of daily varying slope of
c    downstream canal
c
              IF (ivary_slope_daily_opt(IDNCNL(N,IND)) .eq. 1) then
                 call match(cnm(IDNCNL(N,IND)),1,5,canal_name_dvslope,
     &            ncnl_dly_slope,5,imatch_slope)
                 sgn = 1
                 IF (isdnpflw_for_slope(n,ind) .eq. 3) then
                    sgn = -1
                 ENDIF
                 dischg_for_slope_in(imatch_slope,1
     &           ,isdnpflw_for_slope(n,ind)) = dischg_for_slope_in
     &           (imatch_slope,1,isdnpflw_for_slope(n,ind))
     &           + sgn*cout
              ENDIF
c             jstr1=jstr1+1
              IF (flnm(istr(n,ind1)).eq.strname_spec(55)) THEN
                struc_flow_f_dss(ind1) = flws319
              ELSE
                struc_flow_f_dss(ind1) = cout
              END IF
              if (idncnl(n,ind1) .lt. n) then
                 prev_outf_to_ds_cnl(n,idncnl(n,ind1))
     &           = cout
              endif
            END IF
!
   90     CONTINUE
        END IF
        coutd=coutd+cout_asr
c
c
c  compute discharge for canals with daily varying slope
c
        IF (ivary_slope_daily_opt(n) .eq. 1) then
           dischg_for_slope(ncdhd,2,2)
     &     = dischg_for_slope(ncdhd,2,2)
     &     - cout_asr/(2.0*86400.)
           dischg_for_slope(ncdhd,2,3)
     &     = dischg_for_slope(ncdhd,2,3)
     &     + cout_asr/(2.0*86400.)
        ENDIF
```

```
c
c  calculate head drop for selected canals
c
c        if (conv.eq.1) then
c        if (cnm(n) .eq. 'CA3  ' .and. ifyr+iyear-1.eq.1985) then
c======
c        if (cnm(n) .eq.'L8S  ') then
c           write(445,*)N,QU(N)/86400.,TDQU_DN(N),
c    $      QUD/86400.,COUTD/86400.,CHDEP(N),OVLFL/86400.,CGVOL/86400,
c    $      ERROR/86400.,CHSTOR/86400.,HDC(N,ISEAS),ITER,conv
c    $      ,abs(error)/carea,chdep(n)+hdc(n,iseas)/2.0,dplast,acvol
c    $      /86400.,'   IITITITIT  ',month,iday,ifyr+iyear-1
c        endif
c        endif
!
         IF (conv.eq.1.) GO TO 120
         chstor=(chdep(n)+hdc(n,iseas)/2.0-dplast)*carea
         acvol=qu(n)+ovlfl+cgvol-coutd+rfin(n)-etout(n)
         error=chstor-acvol
c        if (cnm(n).eq.'CA3   ') print *,chdep(n),qu(n)/86400.
c    $  ,ovlfl/86400.,cgvol/86400.,coutd/86400.,acvol/86400.
c    $  ,chstor/86400.,error/86400.,'  VOLITER ',hdc(n,iseas)
c    $  ,rfin(n)/86400.,etout(n)/86400.,month,iday,iter
         IF (ivary_slope_daily_opt(n) .eq. 1 .and.
     &  (igrid_loc_upslp(ncdhd) .lt. 0 .or.
     &   pond(igrid_loc_dnslp(ncdhd)) .lt. 0.50)) then
           dischg_for_slope_lower = dischg_for_slope(ncdhd,
     &     2,1) + dischg_for_slope_kn(ncdhd,2,1)
           dischg_for_slope_upper = dischg_for_slope(ncdhd,
     &     1,1) + dischg_for_slope_kn(ncdhd,1,1)
           dischg_for_slope_man = (dischg_for_slope_upper
     &     + dischg_for_slope_lower)/2.0
           dischg_for_slope_man_lat = 0.0
           do mm = 2,3
             dischg_for_slope_man_lat  = dischg_for_slope_man_lat
     &       + (dischg_for_slope(ncdhd,1,mm)
     &       + dischg_for_slope(ncdhd,2,mm))/4.0
     &       + (dischg_for_slope_kn(ncdhd,1,mm)
     &       + dischg_for_slope_kn(ncdhd,2,mm))/4.0
           enddo
           dischg_for_slope_man1 = dischg_for_slope_man
     &     + dischg_for_slope_man_lat
           rmean_dpth_in_canal = chdep(n)+hdc(n,iseas)/2.0
     &     - rmean_cnl_bot_elev(ncdhd)
           cross_area_of_flow = WIDTH(N)*rmean_dpth_in_canal
           wetted_perim = WIDTH(N) + 2.0*rmean_dpth_in_canal
           hyd_radius = cross_area_of_flow/wetted_perim
           if (dischg_for_slope_man1 .gt. 0.000001) then
             sgn = 1.0
           else
             sgn = -1.0
           endif
c          if (conv.eq.1. .and. cnm(n).eq.'C4  ') print *,cnm(n),
c    $     rmean_dpth_in_canal,WIDTH(N),cross_area_of_flow,hyd_radius,
c    $     CL(N)/5280.,dischg_for_slope(ncdhd,1,1),HDC(N,iseas),chdep(n)
c    $     ,dischg_for_slope(ncdhd,1,1),dischg_for_slope_kn(ncdhd,1,1)
c    $     ,dischg_for_slope(ncdhd,2,1),dischg_for_slope_kn(ncdhd,2,1)
```

```
c     $     ,dischg_for_slope(ncdhd,1,2),dischg_for_slope(ncdhd,1,3)
c     $     ,dischg_for_slope(ncdhd,2,2),dischg_for_slope(ncdhd,2,3)
c     $     ,dischg_for_slope_man1,ncdhd
            dischg_for_slope_man_lat_max = 0.0
            dischg_for_slope_man_lat_max = dischg_for_slope_man_lat_max
      $      + (dischg_for_slope_max(ncdhd,1,2)+dischg_for_slope_kn(ncdhd
      $      ,1,2)+dischg_for_slope_max(ncdhd,2,2)+dischg_for_slope_kn
      $      (ncdhd,2,2))*(3./4) - (dischg_for_slope_max(ncdhd,1,3)
      $      +dischg_for_slope_kn(ncdhd,1,3)+dischg_for_slope_max(ncdhd,
      $      2,3)+dischg_for_slope_kn(ncdhd,2,3))*(1./4.)
            dischg_for_slope_max_d = (dischg_for_slope_kn(ncdhd,1,1)
      $     + dischg_for_slope_kn(ncdhd,2,1))/2.0
      $     + dischg_for_slope_max(ncdhd,1,1) +
      $     + dischg_for_slope_man_lat_max

            if (dischg_for_slope_max_d .gt. 0.000001) then
              sgn1 = 1.0
            else
              sgn1 = -1.0
            endif
            head_drop_max_day = sgn1 * (((sgn1*dischg_for_slope_max_d
      $     * flow_resistvty_coef(ncdhd))/(1.49*cross_area_of_flow
      $     *hyd_radius**(2./3.)))** 2.0) * CL(N)
c           if (cnm(n).eq.'C4   ') print *,head_drop_max_day
c     $      ,dischg_for_slope_max_d,dischg_for_slope_kn(ncdhd,1,1)
c     $      ,dischg_for_slope_kn(ncdhd,2,1),dischg_for_slope_max
c     $      (ncdhd,1,1),dischg_for_slope_man_lat_max,
c     $      dischg_for_slope_max(ncdhd,1,2),dischg_for_slope_max
c     $      (ncdhd,1,3),dischg_for_slope_max(ncdhd,2,2),
c     $      dischg_for_slope_max(ncdhd,2,3),'  HD_DROP_MAX'
            HDC(N,iseas) = sgn * (((sgn*dischg_for_slope_man1
      $     * flow_resistvty_coef(ncdhd))/(1.49*cross_area_of_flow
      $     *hyd_radius**(2./3.)))** 2.0) * CL(N)
            head_drop_max = amin1((5.0/52800.)*CL(N),hdrop_max(ncdhd))
            HDC(N,iseas) = sgn*min(sgn*HDC(N,iseas),head_drop_max,
      $     max(head_drop_max_day,0.0))
c           IF (HDC(N,iseas) .gt. -0.0001) THEN
               HDC(N,iseas) = max(HDC(N,iseas),hdrop_min(ncdhd))
c           ENDIF
            diff_hddrop = HDC(N,iseas) - hdrop_prev
            rmax_chg_hdrop = amax1(rmax_dschg_coef(n)-8000.,0.0)
      $     /2000. * 0.15 + 0.5
            if (diff_hddrop .ge. 0.) then
                diff_hddrop = amin1(diff_hddrop,rmax_chg_hdrop)
            else
                sgn = -1.0
                diff_hddrop = amax1(diff_hddrop,sgn*rmax_chg_hdrop)
            endif
            IF (hdrop_prev .gt. HDC(N,iseas) .and. iter.gt. 0) THEN
              HDC(N,iseas) = hdrop_prev - abs(diff_hddrop)
            ELSE
              HDC(N,iseas) = hdrop_prev + abs(diff_hddrop)
            ENDIF
            if (cnm(n).eq.'CA3  ') THEN
               HDC(N,iseas) = max(HDC(N,iseas),0.0)
            endif
            IF (rmax_dschg_coef(n) .gt.10000.) then
```

```
               max_no_iter_stg_var = 60
            ELSE
               max_no_iter_stg_var = 40
            ENDIF
            IF (iter .ge. max_no_iter_stg_var) THEN
               IF (iter .eq. max_no_iter_stg_var) THEN
                 errdpth = abs(error)/carea
                 IF (errdpth .gt. 0.03 .or. cnm(n).eq.'CA3  ') THEN
                   HDC_IT = HDC(N,iseas)
                   if (error .lt. 0.0) then
                     fac_headdrop = 0.25
                   else
                     fac_headdrop = - 0.25
                   endif
                   fac = 0.0
                 ENDIF
               ENDIF
               if (errdpth .gt. 0.03 .or. cnm(n).eq.'CA3  ') THEN
                 if (iter .gt. max_no_iter_stg_var) then
                    HDC_IT = HDC_IT + fac_headdrop
c                   if (nint(dischg_for_slope_man1).gt.0) then
c                     HDC_IT = max(HDC_IT,0.0)
c                   elseif (dischg_for_slope_man1 .lt. -0.0001) then
c                     HDC_IT = min(HDC_IT,0.0)
c                   endif
                 endif
                 HDC(N,iseas) = HDC_IT
               ENDIF
            ENDIF
c           if(cnm(n).eq.'L31NC ') print *,
c    $      HDC(N,iseas),fac_headdrop,'   KLKLKL  ',iter,chdep(n)
c           if (cnm(n).eq.'C11W')
c    $      print *,cnm(n),dischg_for_slope_man1,
c    $      rmean_dpth_in_canal,WIDTH(N),cross_area_of_flow,hyd_radius,
c    $      CL(N)/5280.,dischg_for_slope(ncdhd,1,1),HDC(N,iseas),chdep(n)
c    $      ,dischg_for_slope(ncdhd,1,1),dischg_for_slope_kn(ncdhd,1,1)
c    $      ,dischg_for_slope(ncdhd,2,1),dischg_for_slope_kn(ncdhd,2,1)
c    $      ,dischg_for_slope(ncdhd,1,2),dischg_for_slope(ncdhd,1,3)
c    $      ,dischg_for_slope(ncdhd,2,2),dischg_for_slope(ncdhd,2,3)
c    $     ,dischg_for_slope_kn(ncdhd,1,2),dischg_for_slope_kn(ncdhd,1,3)
c    $     ,dischg_for_slope_kn(ncdhd,2,2),dischg_for_slope_kn(ncdhd,2,3)
c    $      ,ncdhd
          ENDIF
c       if (cnm(n).eq.'PLNTW ')
c    &  print *,dplast,chdep(n)+hdc(n,iseas)/2.0,chstor/86400.,rfin(n)
c    &   /86400.,etout(n)/86400.,acvol/86400.,error/86400.,'  MMMMMMM'
        IF (iter.gt.95.or.(conv.eq.1..and.abs(error)/carea.gt.0.25))
     &   THEN
          PRINT 100, n,cnm(n),iter,chdep(n),qu(n)/86400.,qud/86400.,
     &     coutd/86400.,ovlfl/86400.,cgvol/86400,error/86400.,abs(error)
     &     /carea,month,iday,ifyr+iyear-1,hdc(n,iseas)
        END IF
  100   FORMAT (i3,1x,a5,3x,i3,3x,f15.9,7f13.5,2x,2i3,i5,F6.2)
        IF (abs(error)/carea.lt.0.03.or.iter.gt.110.or.(abs(chdep(n)
     &    -stage_prev) .LT. 1.0E-05 .and. iter .gt. 0 .and.
     &    abs(error)/carea.lt.0.05 .and. ivary_slope_daily_opt(n).eq.0)
     &    .or. (ivary_slope_daily_opt(n).eq.1 .and. iter.gt.40 .and.
```

```
     &    abs(hdc(n,iseas)-hdrop_prev).lt.0.005 .and. abs(error)/carea
     &    .le.0.03) .or.(iter .gt. 95 .and. abs(error)/carea .lt.0.06))
     &    then
          conv=1.
          fac=0.0
          if (ivary_slope_daily_opt(n).eq.1 ) then
            if(iter .gt. max_no_iter_stg_var .and. (errdpth .gt. 0.03
     &      .or. cnm(n).eq.'CA3  ')) HDC(N,iseas) = HDC_IT -fac_headdrop
            if(iter .le. max_no_iter_stg_var) HDC(N,iseas) = hdrop_prev
            if (iter .gt. max_no_iter_stg_var .and. (errdpth .le. 0.03
     &      .and.cnm(n).ne.'CA3  ')) HDC(N,iseas) = hdrop_prev
          endif
c         if (cnm(n).eq.'C13 ') print *,iter,conv,HDC(N,iseas)
c    &      ,HDC_IT,fac_headdrop,'  HJHJHJ  ',chdep(n)
          fac_headdrop=0.0
        ELSE
          IF (error/errprev.lt.0..and.iter.ne.0) THEN
            idp=1
            jdp=1
          END IF
          IF (idp.eq.1) then
            fac=-abs(fac)*0.50*error/abs(error)
          ENDIF
          IF (ivary_slope_daily_opt(n) .eq. 1) THEN
            IF(idp.eq.1.and.iter.gt.max_no_iter_stg_var) fac_headdrop
     $       =-abs(fac_headdrop)*0.5*error/abs(error)
          ENDIF
          IF (idp.eq.0.and.jdp.eq.1) then
            fac=fac*0.98
          ENDIF
          IF (ivary_slope_daily_opt(n) .eq. 1) THEN
            IF (idp.eq.0.and.jdp.eq.1 .and. iter.gt.max_no_iter_stg_var)
     $       fac_headdrop = fac_headdrop *0.98
          ENDIF
c         if (cnm(n).eq.'L8    ') print *,idp
c    $     ,jdp,fac,fac_headdrop
          idp=0
          IF (iter.eq.0.and.error.ge.0.) fac=-fincr
        END IF
!
  110   FORMAT (i4,i3,i4,200f6.2)
!
        stage_prev=chdep(n)
        hdrop_prev = hdc(n,iseas)
c       if (cnm(n).eq.'C51  ') write(*,8888) chdep(n),fac
        chdep(n)=chdep(n)+fac
c       if (cnm(n).eq.'C51  ') write(*,8889) chdep(n)
c8888   format(F12.9,2x,F12.9,'  XXXXXX')
c8889   format(F12.9,'   YYYYYY')
        iter=iter+1
        errprev=error
        GO TO 50
!-----------------------------------------------
c 120   n_canl_rec = 0
c       n_weir_rec = 0
c       if (lcnb(n) .lt. 1 .or. lcnb(n) .gt. 6) then
c         do kk = 1,nout(n)
```

```
c           if (into(n,kk).ne.999.and.into(n,kk).gt.0.and.
c    $      into(n,kk).ne.888) then
c             n_weir_rec = n_weir_rec + 1
c             i_weir_str_idx(n_weir_rec) = kk
c           else
c             adj_for_error_weir(kk) = 0.0
c           endif
c         enddo
c         do not = 1,nstrctr(n)
c           IF (nprty(n,not).ne.0) THEN
c             ind2=nprty(n,not)
c           ELSE
c             ind2=not
c           END IF
c           if (nres(n,ind2).eq.0.and.((IDNCNL(N,ind2).le.nch
c    $      .and. irceive(n,ind2).eq. 1).or.irceive(n,ind2) .gt. 1))
c    $      then
c              n_canl_rec = n_canl_rec + 1
c              i_str_rec_idx(n_canl_rec) = ind2
c           else
c              adj_for_error_struc(ind2) = 0.0
c           endif
c         enddo
c       else
 120      do kk = 1,nout(n)
            adj_for_error_weir(kk) = 0.0
          enddo
          do kk = 1,nstrctr(n)
            adj_for_error_struc(kk) = 0.0
          enddo
c       endif
c       if (abs(error) .le. abs(rfin(n) - etout(n) .or. coutd .lt.
c    $   0.0000001 .or. n_weir_rec+n_canl_rec .eq. 0 ) then
           chdep(n) = chdep(n) - error/carea
c       else
c          chdep(n) = chdep(n) - (etout(n)-rfin(n))/carea
c            - amax1(error - (etout(n)-rfin(n)) - coutd,0.0)
c          coutd = coutd - amin1(error + (etout(n)-rfin(n)),coutd)
c             do kk = 1,n_weir_rec
c                adj_for_error_weir(i_weir_str_idx(kk)) =
c    $           error - (etout(n)-rfin(n))/(n_weir_rec+n_canl_rec)
c                qu(into(n,i_weir_str_idx(kk))) = qu(into(n,
c    $           i_weir_str_idx(kk))) - adj_for_error_weir
c    $           (i_weir_str_idx(kk))
c             enddo
c             do kk = 1,n_canl_rec
c                adj_for_error_struc(i_str_rec_idx(kk)) =
c    $           (error-(etout(n)-rfin(n)))/(n_weir_rec+n_canl_rec)
c                if (irceive(n,i_str_rec_idx(kk)) .eq. 1) then
c                   qu(IDNCNL(N,i_str_rec_idx(KK))) = qu(IDNCNL(N,
c    $              i_str_rec_idx(KK)))-adj_for_error_struc
c    $              (i_str_rec_idx(kk))
c                elseif (irceive(n,i_str_rec_idx(kk)) .eq. 2) then
c                   do il = 1,ncells_rec(n,i_str_rec_idx(KK))
c                      pond(nodrcv(n,i_str_rec_idx(KK),il))
c    $                 = pond(nodrcv(n,i_str_rec_idx(KK),il))
c    $                 - adj_for_error_struc(i_str_rec_idx(kk))
```

```
c    $                 *1.0 /(gdar*(1.0-carfact(nodrcv
c    $                 (n,i_str_rec_idx(KK),il)))*ncells_rec
c    $                 (n,i_str_rec_idx(KK)))
c                   enddo
c                else
c                   do iyy = maxy,1,-1
c                     do node = isum(iyy)+1,isum(iyy+1)
c                      if (ibsn(node).eq.IRVBSN(n,i_str_rec_idx(KK)))
c    $                 then
c                        pond(node)=pond(node)-adj_for_error_struc
c    $                   (i_str_rec_idx(kk))*1.0/(NNDREC(IBSN(NODE))
c    $                   *gdar*(1.0-carfact(node)))
c                      endif
c                     enddo
c                   enddo
c                endif
c              enddo
c         endif
          do kk = 1,nout(n)
             jstr1 = jstr1 + 1
             strflw(jstr1,idss) = weir_struc_f_dss(kk)
     $       - adj_for_error_weir(kk)/86400.
          enddo
          do kk = 1,nstrctr(n)
             IF (nprty(n,kk).ne.0) THEN
                ind2=nprty(n,kk)
             ELSE
                ind2=kk
             END IF
             jstr1 = jstr1 + 1
             strflw(jstr1,idss) = struc_flow_f_dss(ind2)
     $       - adj_for_error_struc(kk)/86400.
c            erradj(n) = erradj(n) + adj_for_error_struc(kk)
c    $        /43560.
          enddo
c
c  check if any outlet structure is discharging for flood control
c  If so all ops for flood control still set to TRUE
c
c  gravity outlets
c
         if (igrav_true_b .ge. n_gated_grav_outlets_fc(n)) THEN
           igrav_true_e = 0
           do istruc = 1,nstrctr(n)
             igrav_true_e_match(istruc) = 0
             do k = 1,no_match_loc(istruc)
                istrucm = imatch_struc_no(istruc,k)
                IF (ncoeff(n,istrucm).eq.1) THEN
                  imgte=1
                ELSE
                  imgte=iseas
                END IF
                if (lgrav(n,istrucm,1) .and. irating(n,istrucm,1,imgte)
     $          .eq.1 .and. ipurpse(n,istrucm,1) .ne. 1) then
                    igrav_true_e = igrav_true_e + 1
                    igrav_true_e_match(istruc) =
     $              igrav_true_e_match(istruc) + 1
```

```
          endif
          if (nsubstr(n,istrucm) .gt. 1) then
             do isb  = 2,nsubstr(n,istrucm)
               if (.not. lgrav(n,istrucm,1).and. irating(n,istrucm
$              ,1,imgte).eq.1 .and. ipurpse(n,istrucm,1).ne.1) then
                  if (lgrav(n,istrucm,isb).and.irating(n,istrucm,
$                 isb,imgte).eq.1 .and.ipurpse(n,istrucm,isb).ne.1)
$                 then
                    igrav_true_e_match(istruc) =
$                   igrav_true_e_match(istruc) + 1
                    igrav_true_e = igrav_true_e + 1
                  endif
                endif
             enddo
          endif
       enddo
     enddo
     do istruc = 1,nstrctr(n)
        IF (imatch_struc_loc(n,istruc) .eq. 0) then
           IF (ncoeff(n,istruc).eq.1) THEN
             imgte=1
           ELSE
             imgte=iseas
           END IF
           if (lgrav(n,istruc,1) .and. irating(n,istruc,1,imgte)
$          .eq.1 .and. ipurpse(n,istruc,1) .ne. 1) then
              igrav_true_e = igrav_true_e + 1
           endif
           if (nsubstr(n,istruc) .gt. 1) then
             do isb  = 2,nsubstr(n,istruc)
               if (.not. lgrav(n,istruc,1).and. irating(n,istruc
$              ,1,imgte).eq.1 .and. ipurpse(n,istruc,1).ne.1) then
                  if (lgrav(n,istruc,isb).and.irating(n,istruc,
$                 isb,imgte).eq.1 .and.ipurpse(n,istruc,isb).ne.1)
$                 then
                    igrav_true_e = igrav_true_e + 1
                  endif
                endif
             enddo
           endif
         ENDIF
      enddo

     do istruc = 1,nstrctr(n)
       if (igrav_true_e .gt. 0) THEN
          IF (igrav_true_e_match(istruc) .eq. 0
$         .and. no_match_loc(istruc) .gt. 0) THEN
             rminstg_match = 9999999.
             do k = 1,no_match_loc(istruc)
                istrucm = imatch_struc_no(istruc,k)
                IF (ncoeff(n,istrucm).eq.1) THEN
                  imgte=1
                ELSE
                  imgte=iseas
                END IF
                do ns = 1,nsubstr(n,istrucm)
                  if (irating(n,istrucm,ns,imgte).eq.1
```

```
     $                 .and. ipurpse(n,istrucm,ns) .ne. 1) then
                         rminstg_match_t = rmnstgs(n,istrucm,iseas)
                         if (rminstg_match_t .lt. rminstg_match) then
                           rminstg_match = rminstg_match_t
                           istrucm_min = istrucm
                         endif
                       endif
                     enddo
                   enddo
c
                   do ns = 1,nsubstr(n,istrucm_min)
                     lgrav(n,istrucm_min,ns) = .TRUE.
                   enddo
                ENDIF
                IF (imatch_struc_loc(n,istruc) .eq. 0) then
                  IF (ncoeff(n,istruc).eq.1) THEN
                     imgte=1
                  ELSE
                     imgte=iseas
                  END IF
                  do isb  = 1,nsubstr(n,istruc)
                   if (irating(n,istruc,isb,imgte).eq.1.and.ipurpse(n,
     $              istruc,isb) .ne. 1) then
                      lgrav(n,istruc,isb) = .TRUE.
                   endif
                  enddo
                endif
              endif
            enddo
          endif
c
c  pump outlets
c
          if (ipump_true_b .ge. n_pump_outlets_fc(n)) then
            ipump_true_e = 0
            do istruc = 1,nstrctr(n)
              ipump_true_e_match(istruc) = 0
              do k = 1,no_match_loc(istruc)
                 istrucm = imatch_struc_no(istruc,k)
                 IF (ncoeff(n,istrucm).eq.1) THEN
                   imgte=1
                 ELSE
                   imgte=iseas
                 END IF
                 if (lpump(n,istrucm,1) .and. irating(n,istrucm,1,imgte)
     $           .eq.2 .and. ipurpse(n,istrucm,1) .ne. 1) then
                     ipump_true_e = ipump_true_e + 1
                     ipump_true_e_match(istruc) =
     $               ipump_true_e_match(istruc) + 1
                 endif
                 if (nsubstr(n,istrucm) .gt. 1) then
                    do isb  = 2,nsubstr(n,istrucm)
                      if (.not. lpump(n,istrucm,1).and. irating(n,istrucm
     $                ,1,imgte).eq.2 .and. ipurpse(n,istrucm,1).ne.1) then
                         if (lpump(n,istrucm,isb).and.irating(n,istrucm,
     $                   isb,imgte).eq.2 .and.ipurpse(n,istrucm,isb).ne.1)
     $                   then
```

```
                    ipump_true_e_match(istruc) =
$                   ipump_true_e_match(istruc) + 1
                    ipump_true_e = ipump_true_e + 1
                  endif
                endif
              enddo
            endif
          enddo
        enddo
        do istruc = 1,nstrctr(n)
           IF (imatch_struc_loc(n,istruc) .eq. 0) then
              IF (ncoeff(n,istruc).eq.1) THEN
                imgte=1
              ELSE
                imgte=iseas
              END IF
              if (lpump(n,istruc,1) .and. irating(n,istruc,1,imgte)
$             .eq.2 .and. ipurpse(n,istruc,1) .ne. 1) then
                 ipump_true_e = ipump_true_e + 1
              endif
              if (nsubstr(n,istruc) .gt. 1) then
                do isb  = 2,nsubstr(n,istruc)
                  if (.not. lpump(n,istruc,1).and. irating(n,istruc
$                 ,1,imgte).eq.2 .and. ipurpse(n,istruc,1).ne.1) then
                    if (lpump(n,istruc,isb).and.irating(n,istruc,
$                   isb,imgte).eq.2 .and.ipurpse(n,istruc,isb).ne.1)
$                   then
                      ipump_true_e = ipump_true_e + 1
                    endif
                  endif
                enddo
              endif
           ENDIF
         enddo
        do istruc = 1,nstrctr(n)
          if (ipump_true_e .gt. 0) THEN
             IF (ipump_true_e_match(istruc) .eq. 0
$            .and. no_match_loc(istruc) .gt. 0) THEN
                rminstg_match = 9999999.
                do k = 1,no_match_loc(istruc)
                   istrucm = imatch_struc_no(istruc,k)
                   IF (ncoeff(n,istrucm).eq.1) THEN
                     imgte=1
                   ELSE
                     imgte=iseas
                   END IF
                   do ns = 1,nsubstr(n,istrucm)
                     if (irating(n,istrucm,ns,imgte).eq.2
$                    .and. ipurpse(n,istrucm,ns) .ne. 1) then
                       rminstg_match_t = rmnstgs(n,istrucm,iseas)
                       if (rminstg_match_t .lt. rminstg_match) then
                         rminstg_match = rminstg_match_t
                         istrucm_min = istrucm
                       endif
                     endif
                   enddo
                enddo
```

```
c
                  do ns = 1,nsubstr(n,istrucm_min)
                    lpump(n,istrucm_min,ns) = .TRUE.
                  enddo
               ENDIF
               IF (imatch_struc_loc(n,istruc) .eq. 0) then
                 IF (ncoeff(n,istruc).eq.1) THEN
                    imgte=1
                 ELSE
                    imgte=iseas
                 END IF
                 do isb  = 1,nsubstr(n,istruc)
                  if (irating(n,istruc,isb,imgte).eq.2.and.ipurpse(n,
     $             istruc,isb) .ne. 1) then
                     lpump(n,istruc,isb) = .TRUE.
                  endif
                 enddo
               endif
            endif
          enddo
        endif
c-----------------------------------------------------------------------------
c
c  compute monthly sums of canal processes
c
        cginfm(n)=cginfm(n)+cgvol/43560.
        qum(n)=qum(n)+(qu(n)+qud)/43560.
        rfinm(n)=rfinm(n)+rfin(n)/43560.
        etoutm(n)=etoutm(n)+etout(n)/43560.
        coutm(n)=coutm(n)+coutd/43560.
        vinm(n)=vinm(n)+(ovlfl/43560.0-tovlf/43560.0)
        dswlm(n)=dswlm(n)+(chdep(n)+hdc(n,iseas)/2.0-dplast)*carea
     $   /43560.
        sumdly=cgvol+qu(n)+qud +ovlfl-tovlf-coutd+rfin(n)-
     &     etout(n)-(chdep(n)+hdc(n,iseas)/2.0-dplast)*carea
        sumdly = sumdly/86400.
c       if (cnm(n).eq.'C13  ') print *,qu(n)/86400.,cgvol/86400.
c    +   ,qud/86400.,rfin(n)/86400.,etout(n)/86400.,coutd/86400.
c    +   ,ovlfl/86400.,tovlf/86400.,(chdep(n)+hdc(n,iseas)/2.0
c    +   -dplast)*carea/86400.,sumdly,'   GGFGFFF  ',chdep(n)
c    +   +hdc(n,iseas)/2.0,dplast,acvol1/86400.,hdc(n,iseas),
c    +   month,iday
        IF (cmx(n).lt.chdep(n)) cmx(n)=chdep(n)
        IF (cmn(n).gt.chdep(n)) cmn(n)=chdep(n)
!-----------------
!-----------------
        prevqu(n)=qu(n)
        qu(n)=0.0
        IF (ivary_slope_daily_opt(n) .eq. 1) then
           do bb = 1,2
            do cc = 1,3
             dischg_for_slope_in(ncdhd,bb,cc) = 0.0
            enddo
           enddo
        ENDIF
        if (cnm(n).eq.'C51W ') out_flow_east0 = 0.0
        if (n.eq.ica3) prev_total_flow_s12 = total_flow_s12
```

```
  130 CONTINUE
!-----------------
      rloc_excess_for_ws(isa1)=rloc_excess_for_ws(isa1)+rlocal_ws_to_e1+
     &rlocal_ws_to_e2+rlocal_ws_to_e3+rlocal_ws_to_es+rlocal_ws_to_ese
!
      prev_totf = totf
      s333ftargprev = s333_reg + s333_env
      s355prevout = kflo(ifls355)
      prev_voldn_s334 = avvol_dns334
      IF (nmc.eq.0) GO TO 140
      WRITE (78,110) ifyr+iyear-1,month,iday,(chdep(mtrc(i))+
     &head_offset(i),i=1,nmc)
      jstrc=0
      DO i=1,nmc
        jstrc=jstrc+1
        cnlstage(jstrc,idss)=chdep(mtrc(i))+head_offset(i)
      END DO
c
!-----------------
!   WRITE OUT THE COMPONENTS OF THE MONTHLY WATER BUDGET FOR THE CANAL
!-----------------
  140 IF (iday.ne.idymax) RETURN
      IF (lp(6)) THEN
        IF (month.eq.1) THEN
          WRITE (82,150) iyear+ifyr-1
  150     FORMAT (////,t20,'CHANNEL FLOW SUMMARY FOR  ',i5,/13x,'------F
     &T. MSL  ------',6x,10('--'),' ACRE FEET ',20('--'),'  ERROR TERM '
     &     ,/4x,' CNL',5x,'CRE',6x,'COUT',6x,'SEEPAGE',6x,'INFLO',7x,'OV
     &LF',7x,'RAIN',7x,'ET',8x,'DWSLY',7x,'DIFF')
        END IF
      END IF
      IF (lp(7)) THEN
        WRITE (72,160) month,iyear+ifyr-1
  160   FORMAT (////,t25,'CHANNEL FLOW SUMMARY FOR  ',i2,'/',i4,/13x,'--
     &---  FT. MSL  -----',6x,10('--'),' ACRE FEET ',20('--'),'  ERROR T
     &ERM ',/4x,' CNL',5x,'CRE',4x,'CSMAX',4x,'CSMIN',6x,'COUT',6x,'SEEP
     &AGE',6x,'INFLO',7x,'OVLF',7x,'RAIN',7x,'ET',8x,'DSWLM',7x,'DIFF')
      END IF
      DO 190 n=1,nch
!------------------------------
! DETERMINE ANNUAL COMPONENTS OF CANAL BUDGET
!-------------------------------
        IF (lp(6)) THEN
          IF (month.eq.1) THEN
            cginfy(n)=0.0
            quy(n)=0.0
            viny(n)=0.0
            dswly(n)=0.0
            couty(n)=0.0
            rfiny(n)=0.0
            etouty(n)=0.0
          END IF
          cginfy(n)=cginfy(n)+cginfm(n)
          quy(n)=quy(n)+qum(n)
          viny(n)=viny(n)+vinm(n)
          dswly(n)=dswly(n)+dswlm(n)
          couty(n)=couty(n)+coutm(n)
```

```
          rfiny(n)=rfiny(n)+rfinm(n)
          etouty(n)=etouty(n)+etoutm(n)
          IF (month.eq.12) THEN
            sumy=cginfy(n)+quy(n)+viny(n)-dswly(n)-couty(n)+rfiny(n)-
     &       etouty(n)
            WRITE (82,180) n,cnm(n),crel(n,1),couty(n),cginfy(n),quy(n),
     &       viny(n),rfiny(n),etouty(n),dswly(n),sumy
          END IF
        END IF
!-----------------------------
!  PRINT OUT MONTHLY COMPONENTS OF CANAL BUDGET
!-----------------------------
        IF (lp(7)) THEN
          sum=cginfm(n)+qum(n)+vinm(n)-dswlm(n)-coutm(n)+rfinm(n)-
     &     etoutm(n)
          WRITE (72,170) n,cnm(n),crel(n,1),cmx(n),cmn(n),coutm(n),
     &     cginfm(n),qum(n),vinm(n),rfinm(n),etoutm(n),dswlm(n),sum
        END IF
  170   FORMAT (i3,2x,a5,f7.1,2f8.2,9f11.0,2f10.1)
  180   FORMAT (i3,2x,a5,f7.1,9f11.0,2f10.1)
!-----------------
  190 CONTINUE
!
      RETURN
      END
```

climvar.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: climvar.inc,v 1.6 2004/01/15 21:27:04 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/climvar.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      common /climvar/
     &s65e_runff_thres(max_no_clim_thres,max_n_lok_zones),
     &trib_rf_et_thres(max_no_clim_thres,max_n_lok_zones),
     &rmulti_seas_thres(max_no_clim_thres,max_n_lok_zones,2),
     &rmulti_seas_pred(12),trib_rf_et,s65e_runff_wkly,
     &multi_seas_pred_opt,
     &trib_rf_et_thres_wca(max_no_clim_thres,max_n_lok_zones),
     &s65e_runff_thres_wca(max_no_clim_thres,max_n_lok_zones),
     &rmulti_seas_thres_wca(max_no_clim_thres,max_n_lok_zones),
     &rmin_clim_indx_thres_wca(max_n_lok_zones,2),
     &trib_rfet_thres,s65e_runff_wkly_thres,
     &rmulti_seas_thres_struc(max_n_ostruc_wca,max_n_wcas),
     &rmulti_seas_thres_chg_sched(max_n_wcas),no_pred_storms_thres_wca,
     &ipulse_level_trib_hyd(max_no_clim_thres,max_n_lok_zones,2),
     &ipulse_level_multi_seas(max_no_clim_thres,max_n_lok_zones,2),
     &itrib_hydro_cond_opt,izone_flex_eaares_opt,
     &iatl_convyr_index,iflex_clim_lokreg_to_calstlres,
     &zoneid(max_n_lok_zones),
     &cgen_trib_hydro_categ(max_no_clim_thres,max_n_lok_zones),
     &cgen_seas_categ(max_no_clim_thres,max_n_lok_zones,2),
     &cgen_multi_seas_categ(max_no_clim_thres,max_n_lok_zones,2),
     &cgen_trib_hydro_categ_wca(max_no_clim_thres,max_n_lok_zones),
     &cgen_seas_categ_wca(max_no_clim_thres,max_n_lok_zones),
     &cgen_multi_seas_categ_wca(max_no_clim_thres,max_n_lok_zones)
C-------------------------------------------------------------------
      CHARACTER zoneid*7,cgen_trib_hydro_categ*7,cgen_seas_categ*7
     $,cgen_multi_seas_categ*7,cgen_trib_hydro_categ_wca*7
     $,cgen_seas_categ_wca*7,cgen_multi_seas_categ_wca*7
```

cnldata.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: cnldata.F,v 1.26 2005/06/10 13:29:02 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/cnldata.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO OPEN AND READ CANAL DATA FILES
C   FILES OPENED:
C    CNDTA22   PHYSICAL PARAMETERS FOR EACH CANAL
C    CANAL22   NODE LOCATIONS TO DEFINE PATH OF EACH CANAL
C    KFLPTS2   KNOWN FLOW SPECIFICATIONS
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE cnldata (crmin, nbranch, istr_count, ires_no,
     & rlist_lvname, import_area_name, rlv_seep_to_canl_c11b,
     & icounty_loc_indx, n_cells, crmin_bel_flr, crmin0, ndsglim,
     & nodedlim, fl_trig_constrt, month_beg_min_mod_flr,
     & month_end_min_mod_flr, iday_beg_min_mod_flr,
     & iday_end_min_mod_flr, node_wd, hdc0, icnl_no_reg_seas,
     & iweir_no_reg_seas, nreg_seas, weir_reg_seas,
     & ivary_slope_daily_opt, mid_point_canl, iwuppflw_for_slope,
     & iwdnpflw_for_slope, isuppflw_for_slope, isdnpflw_for_slope,
     & ilvflw_for_slope, isuppflw_for_slope_kn, isdnpflw_for_slope_kn,
     & is334_open_flg, frac_avail_cap_s333, frac_avail_cap_s334,
     & weir_name, s333s334rg_offset,rindiv_canl_area,
     & no_canl_in_cell,fnm,n_gated_grav_outlets_fc,
     & n_pump_outlets_fc,rmax_dschg_coef,isupp_at_ups_end,
     & iroute_acme_fc_to_sta1e_opt)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
```

```
 INCLUDE 'c2.inc'
 INCLUDE 'stat.inc'
 INCLUDE 'agdata.inc'
 INCLUDE 'wrout.inc'
 INCLUDE 'routc.inc'
 INCLUDE 'simflows.inc'
 INCLUDE 'wcaot.inc'
 INCLUDE 'lakwca.inc'
 INCLUDE 'stas.inc'
 INCLUDE 'environ.inc'
 INCLUDE 'asr_param.inc'
 INCLUDE 'asr1.inc'
 INCLUDE 'wier.inc'
 INCLUDE 'resadj.inc'
 INCLUDE 'specdssout.inc'
 INCLUDE 'resws.inc'
 INCLUDE 'wcasim.inc'
 INCLUDE 'opersched.inc'
 INCLUDE 'canalloc.inc'
 INCLUDE 'lvseepdiv.inc'
 INCLUDE 'loktarg.inc'
 INCLUDE 'add_misc.inc'
 INCLUDE 'stormd.inc'
 INCLUDE 'dualops.inc'
 INCLUDE 'trig_lec_et_param.inc'
 INCLUDE 'lec_et.inc'
 INCLUDE 'cnlneeds.inc'

 DIMENSION colpos_kd(max_n_str_node_dep),
& colpos_ku(max_n_str_node_dep), colpos_sd(max_nostr_cnl),
& colpos_su(max_nostr_cnl), colpos_wd(max_n_weir_outf),
& colpos_wu(max_n_weir_outf), crmin(max_ncnls,max_n_seas),
& crmin_bel_flr(max_ncnls,max_n_seas),ixgr_on_off(50),
& crmin0(max_ncnls,max_n_seas),no_canl_in_cell(max_ncells),
& crmind_bflr(max_n_cnl_min_bel_flr,max_n_seas),
& fl_trig_constrt(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
& hdc0(max_ncnls,max_n_seas),rindiv_canl_area(max_ncells,3),
& icnl_min_mod_indx(max_n_cnl_min_bel_flr),iygr_on_off(50),
& icnl_no_reg_seas(max_n_cnl_w_weir_seas),
& icol_dest_lvseep(max_n_dest_lvsp), icounty_loc_indx(max_ncnls),
& iday_storm(max_n_storm_drwdn),rmax_dschg_coef(max_ncnls),
& ids_trg_struc(max_n_cnl_min_bel_flr),
& ieaarreg(max_n_res), ieaarsc(max_n_res),
& ilvflw_for_slope(max_n_lvsp_loc,2), ires_no(max_n_res),
& irow_dest_lvseep(max_n_dest_lvsp),
& isdnpflw_for_slope(max_ncnls,max_nostr_cnl),
& isdnpflw_for_slope_kn(max_n_str_node_dep),
& isuppflw_for_slope(max_ncnls,max_nostr_cnl),
& isuppflw_for_slope_kn(max_n_str_node_dep),
& ivary_slope_daily_opt(max_ncnls),
& iwdnpflw_for_slope(max_ncnls,max_n_weir_outf),
& iweir_no_reg_seas(max_n_cnl_w_weir_seas),
& iwuppflw_for_slope(max_ncnls,max_n_weir_outf),
& ixglim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
& iyear_storm(max_n_storm_drwdn),
& iyglim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
& lev_no(max_no_lvsp), mid_point_canl(max_ncnls),
```

```
     & month_storm(max_n_storm_drwdn), n_cells(max_n_gage_loc_outp),
     & nbranch(max_ncnls), ndsglim(max_ncnls,max_nostr_cnl),
     & no_of_down_strm_loc(max_n_str_lok_in_wca),
     & no_of_stg_outp(max_n_gage_loc_outp),
     & node_wd(max_ncnls,max_n_weir_outf),
     & nodedlim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
     & nprstrmn(max_ncnls),carea_in_cell(max_ncells),
     & pump_effic(max_ncnls,max_nostr_cnl,max_n_substr), reg(12),
     & rowpos_kd(max_n_str_node_dep), rowpos_ku(max_n_str_node_dep),
     & rowpos_sd(max_nostr_cnl), rowpos_su(max_nostr_cnl),
     & rowpos_wd(max_n_weir_outf), rowpos_wu(max_n_weir_outf),
     & weir_reg_seas(max_n_cnl_w_weir_seas,max_n_weir_outf,max_n_seas),
     & outf_struc_names_on_off(max_nostr_cnl),
     & n_gated_grav_outlets_fc(max_ncnls),n_pump_outlets_fc(max_ncnls),
     & isupp_at_ups_end(max_ncnls,max_nostr_cnl)

      CHARACTER borrow_cnl_name_dnstrm*5, borrow_cnl_name_upstrm*5,
     & canal_dest_name_dual_ops*5, canal_name_drawdown*5,
     & canal_name_dual_ops*5, canal_name_reg_seas*5, canl_id*5,
     & canl_id1*5, canl_id2*5, canl_id3*5,str_name_chg_loc_fc_ops*6,
     & cidncnl(max_ncnls,max_nostr_cnl)*5,cnl_name_upstrm_fc_ops*5,
     & cinto(max_ncnls,max_n_weir_outf)*5,cnl_name_trg_ops*5,
     & cnl_name_fc(max_n_ds_cnl_lim)*5, cnme*5,
     & county_loc*2, canl_min_mod_name*5,str_name_on_off*6,
     & dest_canal_name_lvseep(max_n_dest_lvsp)*5,
     & ds_canal_name_lim(max_ncnls,max_n_ds_cnl_lim)*5,
     & ds_targ_id(max_n_str_w_ds_targ)*6,
     & import_area_name(max_n_targ_area)*5,canal_name_on_off*5,
     & name_of_inflow_struc_res(max_ncnls,max_nostr_cnl,
     &   max_n_res_rec_str_flw)*6, namelok*5,
     & outf_struc_names(max_nostr_cnl)*6,char_id_cnl_ws_resasr*5,
     & outf_struc_names_res(max_nostr_res)*6,
     & res_name_cnl*6, res_name_drawdown*6,
     & res_name_frm_cnl(max_ncnls,max_nostr_cnl,
     &   max_n_res_rec_str_flw)*6, res_struc_inf_ref(max_nostr_res)*6,
     & rlist_lvname(max_no_lvsp)*5, rlv_seep_to_canl_c11b*5,
     & str_id(max_ncnls,max_nostr_cnl)*6, struc_name_reg_seas*6,
     & struc_tvary_name*6, targ_name_byp_res(max_n_res_rec_str_flw)*5,
     & upstrm_canal_name(max_ncnls)*5,outf_struc_names_on_off*6,
     & weir_name(max_ncnls,max_n_weir_outf)*6,
     & weir_name_reg_seas(max_n_weir_outf)*6,
     & outf_struc_names_ops(max_nostr_cnl)*6,
     & fnm(max_n_str_node_dep)*6,ups_canal_name_on_off*5,
     & str_name_alt_loc_trg_ops*6,cnl_name_alt_trg_ops*5

      INTEGER colpos_kd, colpos_ku, colpos_sd, colpos_su, colpos_wd,
     & colpos_wu, rowpos_kd, rowpos_ku, rowpos_sd, rowpos_su, rowpos_wd,
     & rowpos_wu

      REAL lvseep_divers, lvseep_pump_cap

      DOUBLE PRECISION CRMIN,CRMIN0,hdc0,crmin_bel_flr
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--

C  READ CANAL SPECIFICATION DATA

      DO I=1,NCH+1
```

```
         QU(I) = 0.0
      END DO

      do node = 1,max_ncells
         carfact(node) = 0.0
         carea_in_cell(node) = 0.0
         no_canl_in_cell(node) = 0.0
      enddo
      NSTR = 0
      iroute_to_miamic_via_g200a = 0
      READ (11,*) no_of_struc_w_ds_targ_no_res
      DO k = 1,no_of_struc_w_ds_targ_no_res
         READ (11,*) struc_name_w_targ (k),
     $   ds_targ_id(k),frc_flow_targ(k)
      ENDDO

      IF (no_of_struc_w_ds_targ_no_res .GT. 0) THEN
         DO kk = 1, no_of_struc_w_ds_targ_no_res
            CALL match(ds_targ_id(kk),1,5,import_area_name ,
     $      ntotal_no_import_areas+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,10) ds_targ_id(kk),struc_name_w_targ(kk)
               STOP
            ENDIF
            ids_trg_struc(kk) = imatch
         END DO
      ENDIF
      READ (11,'(i5,2x,20(A6,1x),5(/7X,20(A6,1x)))') no_struc_spec ,
     $(strname_spec(i),i=1,no_struc_spec)
      READ (11,*)
      DO 190 N = 1,NCH
         n_gated_grav_outlets_fc(n) = 0
         n_pump_outlets_fc(n) = 0
         READ (11,20) CNME,(HDC(N,I),I=1,2),WIDTH(N),CHHC(N),CRMIN(N,1),
     $   CRMIN(N,2),BEGSTG(N),upstrm_canal_name(n),NBRANCH(N),LCNB(N),
     $   NOUT(N),iserv_area_loc(n),county_loc
         IF (county_loc .EQ. 'PB') THEN
            icounty_loc_indx(n) = 1
         ELSE IF (county_loc .EQ. 'BR') THEN
            icounty_loc_indx(n) = 2
         ELSE IF (county_loc .EQ. 'DA') THEN
            icounty_loc_indx(n) = 3
         ELSE
            icounty_loc_indx(n) = 0
         ENDIF
         DO iseason = 1,2
            CRMIN_bel_flr(N,iseason) = CRMIN(N,iseason)
            CRMIN0(N,iseason) = CRMIN(N,iseason)
            HDC0(N,iseason) = HDC(N,iseason)
         ENDDO
c
c  check if order of canals input is consistent with canal location file
c
         IF (CNME .NE. CNM(N)) THEN
             WRITE(*,3) CNME,CNM(N)
             STOP
         ENDIF
```

```
c
c  input structure specifications
c
         DO i = 1,nout(n)
            READ (11,30) weir_name(n,i),CINTO(N,I),COLPOS_WU(I),
     $      ROWPOS_WU(I),COLPOS_WD(I),ROWPOS_WD(I),STRCF(N,i),CREL(N,i),
     $      GWDTH(N,i)
            CREL2(N,i) = CREL(N,i)
            IF (COLPOS_WD(I) .GT. 0 .AND. ROWPOS_WD(I) .GT. 0)
     $      NODE_WD(N,I) = COLPOS_WD(I)-MINX(ROWPOS_WD(I))+1
     $      +ISUM(ROWPOS_WD(I))
         END DO
         READ (11,'(3X,I3,6I5)') NTIDEOUT(N),(YCNLNODE(N,I),I = 1,
     $   NTIDEOUT(N)),NSTRCTR(N)

         DO i = 1,NSTRCTR(N)
            imatch_struc_loc(n,i) = 0
            iopt_for_interdep_on_off(n,i) = 0
         ENDDO
c
         DO i = 1,nstrctr(n)
c
            ides_cap_limit(n,i) = 0
            DO m = 1,12
               frac_struc_cap(n,i,m) = 1.0
            END DO
c
            READ (11,'(A6)') str_id(N,I)
            IF (str_id(N,I) .EQ. strname_spec(29))
     $      iroute_to_miamic_via_g200a = 1
            READ (11,'(5I6)') colpos_su(i),rowpos_su(i),colpos_sd(i) ,
     $      rowpos_sd(i)
c
            if (rowpos_su(i) .gt. 0 .and. colpos_su(i) .gt. 0)
     +       then
               icell_no_su(n,i) = colpos_su(i) - minx(rowpos_su(i))
     +         + 1 + isum(rowpos_su(i))
            else
               icell_no_su(n,i) = -901
            endif
c
            if (rowpos_sd(i) .gt. 0 .and. colpos_sd(i) .gt. 0)
     +       then
               icell_no_sd(n,i) = colpos_sd(i) - minx(rowpos_sd(i))
     +         + 1 + isum(rowpos_sd(i))
            else
               icell_no_sd(n,i) = -901
            endif
c
            READ (11,'(I6)') ISPEC(N,I)
            READ (11,'(I6)') IRCEIVE(N,I)
            IRCEIVE2(N,I) = IRCEIVE(N,I)
C
            IF (IRCEIVE(N,I) .EQ. 1) THEN
               READ (11,'(A5)') CIDNCNL(N,I)
            ELSEIF (IRCEIVE(N,I) .EQ.  2) THEN
               READ (11,'(5X,I3,20I5)') ncells_rec(N,I) ,(IXRCV(N,i,j),
```

```
     $         IYRCV(N,I,j) ,j = 1,ncells_rec(N,I))

               DO j = 1,ncells_rec(N,I)
                  NODRCV(N,I,j) = IXRCV(N,i,j) - MINX(IYRCV(N,i,j)) +  1
     $            + ISUM(IYRCV(N,i,j))
                  NODRCV2(N,I,j) = NODRCV(N,I,j)
               END DO
               CIDNCNL(N,I) = 'NOCNL'
            ELSEIF (IRCEIVE(N,I) .EQ. 3) THEN
               READ (11,'(5X,10I5)') IRVBSN(N,I)
               CIDNCNL(N,I) = 'NOCNL'
            ENDIF
c
c
            READ (11,'(I6)') NPRTY(n,i)
            READ (11,'(5F6.2)') (RMXSTG(N,I,J),J=1,2)
            READ (11,'(5F6.2)') (RMNSTGS(N,I,J),J=1,2)
            DO J = 1,2
               RMXSTG2(N,I,J) = RMXSTG(N,I,J)
               RMNSTG2(N,I,J) = RMNSTGS(N,I,J)
               RMXSTG0(N,I,J) = RMXSTG(N,I,J)
               RMNSTG0(N,I,J) = RMNSTGS(N,I,J)
            END DO
c
c calculate depth above minimum for reduction of capacity for flood control
c
            DO J=1,2
              dpth_abve_mn = 0.5 * (RMXSTG(N,I,J)-RMNSTGS(N,I,J))
              dpth_above_mn = amin1(dpth_abve_mn,0.2)
              dpth_above_mn_cap_red(n,i,j) = max(dpth_above_mn,0.1)
            ENDDO
c
            READ (11,'(F6.2,1x,i2,1x,i2,5(2i3,F6.2))') RMINHD(N,I),
     $      ides_cap_limit(n,i),ndsglim(n,i),(ixglim(n,i,j),
     $      iyglim(n,i,j),fl_trig_constrt(n,i,j),j=1,ndsglim(n,i))
c
            do j=1,ndsglim(n,i)
               nodedlim(n,i,j) = ixglim(n,i,j) - minx(iyglim(n,i,j))
     $         + 1 + isum(iyglim(n,i,j))
            enddo
c
            ids_trg_idx(n,i) = ntotal_no_import_areas+1
            IF (no_of_struc_w_ds_targ_no_res .GT. 0) THEN
               DO kk = 1,no_of_struc_w_ds_targ_no_res
                  IF (str_id(N,I) .EQ. struc_name_w_targ(kk)) THEN
                     ids_trg_idx(n,i) = ids_trg_struc(kk)
                     GO TO 40
                  ENDIF
               END DO
            ENDIF
   40       READ (11,'(I6,2x,5(A6,2x,f5.2,2x,A5,2x,i1,2x,A6,2x))') NRES
     $      (N,I) ,(res_name_frm_cnl(n,i,ir),pctres(n,i,ir) ,
     $      targ_name_byp_res(ir),iopt_to_deiver_bel_targ(n,i,ir) ,
     $      name_of_inflow_struc_res(n,i,ir),ir = 1,nres(n,i))
            READ (11,'(I6,2x,5(A5,2X,F7.2,2X))') NDSCLIM(N,I) ,
     $      (ds_canal_name_lim(n,j) ,CNLSTG(N,I,J), J = 1,NDSCLIM(N,I))
            READ (11,'(I6)') NCOEFF(N,I)
```

```
c
            DO J = 1,NDSCLIM(N,I)
              CNLSTG2(N,I,J) = CNLSTG(N,I,J)
            ENDDO
c
            IF (NRES(N,I) .GT. 0) THEN
               DO ir = 1,NRES(N,I)
                  res_name_cnl = res_name_frm_cnl(n,i,ir)
                  CALL match(res_name_cnl,1,6,resname,ntotres+1,6,
     $            imatch)
                  IF (imatch .EQ. 0) THEN
                     WRITE(*,50) res_name_cnl,str_id(n,i)
                  ENDIF
                  irsrcvr(n,i,ir) = imatch
                  DO kk = 1,ninstr(irsrcvr(n,i,ir))
                     res_struc_inf_ref(kk) = res_struc_inflow_name
     $               (irsrcvr(n,i ,ir),kk)
                  END DO
                  CALL match(targ_name_byp_res(ir),1,5,
     $            import_area_name ,ntotal_no_import_areas+1,5,imatch)
                  IF (imatch .EQ. 0) THEN
                     WRITE(*,60) targ_name_byp_res(ir),str_id(n,i)
                  ENDIF
                  itrg_byp_r_idx(n,i,ir) = imatch
                  IF (pctres(n,i,ir) .lt. 0.010) THEN
                     CALL match(name_of_inflow_struc_res(n,i,ir),1,6,
     $               res_struc_inf_ref,ninstr(irsrcvr(n,i,ir)),6,imatch)
                     IF (imatch .EQ. 0) THEN
                        WRITE(*,70) name_of_inflow_struc_res(n,i,ir),
     $                  str_id(n,i)
                     ENDIF
                     i_res_inf_struc_indx(n,i,ir) = imatch
                  ENDIF
               END DO
            ENDIF
            IF (NDSCLIM(N,I) .GT. 0) THEN
               DO j = 1,NDSCLIM(N,I)
                  CALL match(ds_canal_name_lim(n,j),1,5,cnm,nch+1,5,
     $            imatch)
                  IF (imatch .EQ. 0) THEN
                     WRITE(*,80) ds_canal_name_lim(n,j)
                     STOP
                  ENDIF
                  ICNUM(N,I,J) = imatch
               END DO
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
            IF (str_id(n,i) .EQ. strname_spec(67) .OR. str_id(n,i) .EQ.
     $      strname_spec(51) .OR. str_id(n,i) .EQ. strname_spec(50))
     $      THEN
               READ (11,'(I6,2x,5(F6.2,1x))') n_trigger_fc(n,i) ,
     $         (stg_trigger_fl(n,i,j),j=1,n_trigger_fc(n,i))
               READ (11,'(I6,2x,5(F6.2,1x))') n_stg_min_cnl(n,i) ,
     $         (stg_min_fc(n,i,j),j=1,n_stg_min_cnl(n,i))
               IF (str_id(n,i) .EQ. strname_spec(50)) THEN
                  READ (11,*) frac_reg_flow_nesrs_s355
```

```
               ELSE IF(str_id(n,i) .EQ. strname_spec(51)) THEN
                  READ (11,*) frac_reg_flow_nesrs_s333 ,
     $            iopt_for_reg_releases_s334,s333s334rg_offset
                  READ (11,*) is334_open_flg,frac_avail_cap_s333,
     $            frac_avail_cap_s334
               ENDIF
            ENDIF
            IF (str_id(n,i) .EQ. strname_spec(68)) THEN
               READ (11,'(5x,A3)') earth_plug_opt
               READ (11,'(I6,2x,5(A5,1x,I3,2x))') n_trigger_fc_cnl
     $         (n,i) ,(cnl_name_fc(j),n_trig_for_oper(n,i,j) ,j=1,
     $         n_trigger_fc_cnl(n,i))
               DO j = 1,n_trigger_fc_cnl(n,i)
                  READ (11,'(5(F6.2,1x))') (stg_trig_oper(n,i,j,k),k =
     $            1, n_trig_for_oper(n,i,j))
                  CALL match (cnl_name_fc(j),1,5,cnm,nch+1,5,imatch)
                  IF (imatch .EQ. 0) THEN
                     WRITE(*,90) cnl_name_fc(j)
                  ENDIF
                  icnl_no_fc(n,i,j) = imatch
               END DO
            ENDIF
            IF (str_id(n,i) .EQ. strname_spec(95)) THEN
                READ (11,*) iroute_acme_fc_to_sta1e_opt
            ENDIF
C
            READ (11,'(i6)') NSUBSTR(N,I)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c           ngrav_fc_subs = 0
c           npump_fc_subs = 0
            DO NS = 1,NSUBSTR(N,I)
               READ (11,*) IPURPSE(N,I,NS) ,(IRATING(N,I,NS,J),STRDCAP
     $         (N,I,NS,J),DISCOEF (N,I,NS,J),PWR(N,I,NS,J),STGINTKE
     $         (N,I,NS,J),J = 1,NCOEFF(N,I))
               IF (IRATING(N,I,NS,1) .EQ. 2) THEN
                  IF (str_id(n,i) .EQ. strname_spec(12)) THEN
                     READ (11,'(I6,1x,F6.2,2x,A3,I3)')INGVHDOPT(N,I,NS),
     $               pump_effic(n,i,ns),back_sp_pump_s9_opt,
     $               iopt_for_interdep_on_off(n,i)
                  ELSEIF (str_id(n,i) .EQ. strname_spec(31)) THEN
                     READ (11,'(I6,1x,F6.2,2I3)') INGVHDOPT(N,I,NS),
     $               pump_effic(n,i,ns),iseep_lim_s356_opt,
     $               iopt_for_interdep_on_off(n,i)
                  ELSE
                     READ (11,'(I6,1x,F6.2,I3)') INGVHDOPT(N,I,NS),
     $               pump_effic(n,i,ns),iopt_for_interdep_on_off(n,i)
                  ENDIF
                  DO KI = 1,NCOEFF(N,I)
                     DISCOEF(N,I,NS,KI) = DISCOEF(N,I,NS,KI) *
     $               pump_effic(n,i,ns)
                  END DO
                ENDIF
            END DO
         END DO

c
c
```

```
         DO i = 1,nout(n)
            IF (CINTO(N,I) .NE. 'LAND ') THEN
               CALL match(CINTO(N,I),1,5,cnm,nch+1,5,imatch)
               IF (imatch .EQ. 0 .AND. CINTO(N,I) .NE. 'OCEAN' .AND.
     $         CINTO(N,I) .NE. 'BNDRY') THEN
                  WRITE(*,100) CINTO(N,I),weir_name(n,i)
                  STOP
               ENDIF
               into(n,i) = imatch
               IF (CINTO(N,I) .EQ. 'BNDRY') into(n,i) = 888
            ELSE
               into(n,i) = 999
            ENDIF
         END DO
c

         DO I = 1,NOUT(N)
            kmatch_cnl_loc_weir_up = 0
            kmatch_cnl_loc_weir_dn = 0
            DO IPOS = 1,NODCR(N)
               IF (COLPOS_WU(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_WU(I) .EQ.
     $         YCN(N,IPOS)) THEN
                  NODEPOS_WU(N,I) = IPOS
                  IF (NODEPOS_WU(N,I) .eq. NODCR(N)) THEN
                     iwuppflw_for_slope(n,i) = 1
                  ELSEIF (NODEPOS_WU(N,I) .le. mid_point_canl(N)) THEN
                     iwuppflw_for_slope(n,i) = 2
                  ELSEIF (NODEPOS_WU(N,I) .gt. mid_point_canl(N)
     $              .AND. NODEPOS_WU(N,I) .ne. NODCR(N)) THEN
                     iwuppflw_for_slope(n,i) = 3
                  ENDIF
                  kmatch_cnl_loc_weir_up = 1
                  GO TO 110
               ENDIF
            END DO
  110       IF (CINTO(N,I) .NE. 'LAND '.AND. CINTO(N,I) .NE. 'OCEAN'
     $      .AND. CINTO(N,I) .NE. 'BNDRY') THEN
               DO IPOS = 1,NODCR(into(n,i))
                  IF (COLPOS_WD(I) .EQ. XCN(into(n,i),ipos) .AND.
     $            rowpos_wd(i) .EQ. ycn(into(n,i),ipos)) THEN
                     nodepos_wd(n,i) = ipos
                     IF (NODEPOS_wd(n,i) .eq. 1) THEN
                       iwdnpflw_for_slope(n,i) = 1
                     ELSEIF (NODEPOS_wd(n,i) .le. mid_point_canl
     $               (into(n,i)).and.NODEPOS_WD(n,i) .ne.1) THEN
                       iwdnpflw_for_slope(n,i) = 2
                     ELSEIF (NODEPOS_wd(n,i) .gt. mid_point_canl
     $               (into(n,i))) THEN
                       iwdnpflw_for_slope(n,i) = 3
                     ENDIF
                     kmatch_cnl_loc_weir_dn = 1
                     GO TO 120
                  ENDIF
               END DO
            ELSE
               nodepos_wd(n,i) = -901
```

```
               kmatch_cnl_loc_weir_dn = 1
            ENDIF
  120       CONTINUE
            IF (kmatch_cnl_loc_weir_up .EQ. 0) THEN
                WRITE(*,650) COLPOS_WU(I),ROWPOS_WU(I),cnm(n)
                STOP
            ENDIF
            IF (kmatch_cnl_loc_weir_dn .EQ. 0) THEN
                WRITE(*,750) COLPOS_WD(I),rowpos_wd(i),cnm(n)
                STOP
            ENDIF
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         DO i = 1,nstrctr(n)
            CALL match(str_id(N,I),1,6,flnm,300,6,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,140) str_id(N,I),cnm(n)
               STOP
            ENDIF
            istr(n,i) = imatch
         END DO
         DO i = 1,nstrctr(n)
            CALL match(CIDNCNL(N,I),1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,150) cidncnl(n,i),cnm(n),str_id(n,i)
               STOP
            ENDIF
            idncnl(n,i) = imatch
            kmatch_cnl_loc_up = 0
            DO kk = 1,NODCR(N)
                IF (colpos_su(i) .EQ. xcn(n,kk) .AND. rowpos_su(i)
     $          .EQ. ycn(n,kk)) THEN
                    kmatch_cnl_loc_up = 1
                ENDIF
            ENDDO
            IF (kmatch_cnl_loc_up .EQ. 0) THEN
                WRITE(*,850) colpos_su(i),rowpos_su(i),flnm(istr(n,i))
                STOP
            ENDIF
            kmatch_cnl_loc_dn = 0
            IF (idncnl(n,i) .LE. nch) THEN
               DO kk = 1,NODCR(idncnl(n,i))
                  IF (colpos_sd(i) .EQ. xcn(idncnl(n,i),kk) .AND.
     $            rowpos_sd(i) .EQ. ycn(idncnl(n,i),kk)) THEN
                     kmatch_cnl_loc_dn = 1
                  ENDIF
               ENDDO
            ENDIF
            IF (kmatch_cnl_loc_dn.EQ.0 .AND. idncnl(n,i).LE.nch) THEN
                WRITE(*,950) colpos_sd(i),rowpos_sd(i),flnm(istr(n,i))
                STOP
            ENDIF
            IDNCNL2(n,i) = idncnl(n,i)
            NO_DS_CNL_WS(N,I) = 1
            IDNCNL_WS(N,I,1) = IDNCNL(N,I)
         END DO
c
```

```
c
         DO i = 1,nstrctr(n)
c
c
            kk = i + 1
            if (imatch_struc_loc(n,i) .eq. 0) then
               do k = kk,nstrctr(n)
                  if (imatch_struc_loc(n,k) .eq. 0) then
                    if (icell_no_su(n,i).eq.icell_no_su(n,k) .AND.
     $              irceive(n,i).eq.irceive(n,k) .AND. ((irceive(n,i)
     $              .eq. 2 .and. nodrcv(n,i,1).eq.nodrcv(n,k,1))
     $              .or. (irceive(n,i).eq.1 .and. idncnl(n,i) .eq.
     $              idncnl(n,k) .and. icell_no_sd(n,i) .eq.
     $              icell_no_sd(n,k)))) THEN
                       imatch_struc_loc(n,i)=i
                       imatch_struc_loc(n,k)=i
                       if (k .eq. kk) then
                          no_match_loc(i) = 2
                          imatch_struc_no(i,1) = i
                       else
                          no_match_loc(i) = no_match_loc(i) + 1
                       endif
                       imatch_struc_no(i,no_match_loc(i)) = k
                    endif
                  endif
               enddo
            endif
            DO ipos = 1,nodcr(n)
               IF (COLPOS_SU(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_SU(I)
     $         .EQ. YCN(N,IPOS)) THEN
                  NODEPOS_SU(N,I) = IPOS
                  IF (NODEPOS_SU(N,I) .eq. NODCR(N) .OR. flnm(istr(n,i))
     $            .EQ. 'S12A  ' .OR. flnm(istr(n,i)) .EQ. 'S12B  ' .OR.
     $            flnm(istr(n,i)) .EQ. 'S12C  ' .OR. flnm(istr(n,i))
     $            .EQ. 'S12D  ' .OR. flnm(istr(n,i)) .EQ. 'S333  ') THEN
                     isuppflw_for_slope(n,i) = 1
                     isupp_at_ups_end(n,i) = 0
                  ELSEIF (NODEPOS_SU(N,I) .le. mid_point_canl(N)) THEN
                    isuppflw_for_slope(n,i) = 2
                    isupp_at_ups_end(n,i) = 0
                    IF (NODEPOS_SU(N,I) .eq. 1) THEN
                       IF (ICL(n,ipos) .eq. 1) THEN
                         IF (COLPOS_SD(I) .ne. COLPOS_SU(I)) THEN
                            isuppflw_for_slope(n,i) = 1
                            isupp_at_ups_end(n,i) = 1
                         ENDIF
                       ELSEIF (ICL(n,ipos) .eq. 2) THEN
                         IF (ROWPOS_SD(I) .ne. ROWPOS_SU(I)) THEN
                            isuppflw_for_slope(n,i) = 1
                            isupp_at_ups_end(n,i) = 1
                         ENDIF
                       ELSE
                         isuppflw_for_slope(n,i) = 1
                         isupp_at_ups_end(n,i) = 1
                       ENDIF
                    ENDIF
                  ELSEIF (NODEPOS_SU(N,I) .gt. mid_point_canl(N)
```

```
     $              .AND. NODEPOS_SU(N,I) .ne. NODCR(N)) THEN
                     isuppflw_for_slope(n,i) = 3
                     isupp_at_ups_end(n,i) = 0
                  ENDIF
                  GO TO 160
               ENDIF
            END DO
  160       IF (idncnl(n,i) .le. nch) THEN
               DO IPOS = 1,NODCR(idncnl(n,i))
                  IF (idncnl(n,i) .le. nch) THEN
                     IF (COLPOS_SD(I) .EQ. XCN(idncnl(n,i),ipos) .AND.
     $               rowpos_sd(i) .EQ. ycn(idncnl(n,i),ipos)) THEN
                        nodepos_sd(n,i) = ipos
                        IF (NODEPOS_sd(n,i) .eq. 1) THEN
                          isdnpflw_for_slope(n,i) = 1
                        ELSEIF (NODEPOS_sd(n,i) .le. mid_point_canl
     $                  (idncnl(n,i)).and.NODEPOS_sd(n,i) .ne. 1) THEN
                          isdnpflw_for_slope(n,i) = 2
                        ELSEIF (NODEPOS_sd(n,i) .gt. mid_point_canl
     $                  (idncnl(n,i)))  THEN
                          isdnpflw_for_slope(n,i) = 3
                        ENDIF
                        IF (i .lt. nstrctr(n)) THEN
                           GO TO 170
                        ELSE
                           GO TO 180
                        ENDIF
                     ENDIF
                  ENDIF
               END DO
            ELSE
               nodepos_sd(n,i) = -901
            ENDIF
  170       CONTINUE
         END DO
c
  180    rmax_dschg_coef(n) = -99999.
         DO i = 1,nstrctr(n)
            ntot_pump_fc_subs = 0
            ntot_grav_fc_subs = 0
            DO ns = 1,nsubstr(n,i)
               if (discoef(n,i,ns,1).gt.rmax_dschg_coef(n))
     $          rmax_dschg_coef(n) = discoef(n,i,ns,1)
            ENDDO
            DO k = 1,no_match_loc(i)
c
c    outlet structures whose source and destination locations match
c
               npump_fc_subs = 0
               ngrav_fc_subs = 0
               DO ns = 1,nsubstr(n,imatch_struc_no(i,k))
                  IF (irating(n,imatch_struc_no(i,k),ns,1).eq.2) THEN
c
c    pumps that match
c
                     IF (ns .eq. 1) then
                        IF (ipurpse(n,imatch_struc_no(i,k),ns) .ne. 1)
```

```
     $                  THEN
                            npump_fc_subs = npump_fc_subs + 1
                            ntot_pump_fc_subs = ntot_pump_fc_subs + 1
                        ENDIF
                     ELSE
                        IF (ipurpse(n,imatch_struc_no(i,k),ns) .ne. 1)
     $                  THEN
                            IF (npump_fc_subs .eq. 0) THEN
                               ntot_pump_fc_subs = ntot_pump_fc_subs + 1
                            ENDIF
                            npump_fc_subs = npump_fc_subs + 1
                        ENDIF
                     ENDIF
                     IF (ntot_pump_fc_subs .eq. 1) n_pump_outlets_fc(n)
     $               = n_pump_outlets_fc(n) + 1
                  ELSE
c
c    gravity structures that match
c
                     IF (irating(n,imatch_struc_no(i,k),ns,1).eq.1) THEN
                        IF (ns .eq. 1) THEN
                           IF (ipurpse(n,imatch_struc_no(i,k),ns).ne.1)
     $                     THEN
                              ngrav_fc_subs = ngrav_fc_subs + 1
                              ntot_grav_fc_subs = ntot_grav_fc_subs + 1
                           ENDIF
                        ELSE
                           IF (ipurpse(n,imatch_struc_no(i,k),ns).ne.1)
     $                     THEN
                              IF (ngrav_fc_subs .eq. 0) THEN
                                 ntot_grav_fc_subs=ntot_grav_fc_subs+1
                              ENDIF
                              ngrav_fc_subs = ngrav_fc_subs + 1
                           ENDIF
                        ENDIF
                        IF (ntot_grav_fc_subs .eq. 1)
     $                   n_gated_grav_outlets_fc(n) =
     $                   n_gated_grav_outlets_fc(n) + 1
                     ENDIF
                  ENDIF
               ENDDO
            ENDDO
         ENDDO
c
         DO i = 1,nstrctr(n)
            IF ( imatch_struc_loc(n,i) .eq. 0) THEN
c
c    outlet structures whose source and destination locations do NOT match
c
               npump_fc_subs = 0
               ngrav_fc_subs = 0
               DO ns = 1,nsubstr(n,i)
                  IF (irating(n,i,ns,1).eq.2) THEN
c
c    pumps that do NOT match
c
                     IF (ns .eq. 1) then
```

```
                        IF (ipurpse(n,i,ns) .ne. 1)
     $                  THEN
                            npump_fc_subs = npump_fc_subs + 1
                            n_pump_outlets_fc(n) = n_pump_outlets_fc(n)
     $                      + 1
                        ENDIF
                     ELSE
                        IF (ipurpse(n,i,ns) .ne. 1)
     $                  THEN
                            IF (npump_fc_subs .eq. 0) THEN
                               n_pump_outlets_fc(n)=n_pump_outlets_fc(n)
     $                          + 1
                            ENDIF
                            npump_fc_subs = npump_fc_subs + 1
                        ENDIF
                     ENDIF
                  ELSE
c
c    gravity structures that do NOT match
c
                     IF (irating(n,i,ns,1).eq.1) THEN
                        IF (ns .eq. 1) THEN
                           IF (ipurpse(n,i,ns).ne.1)
     $                     THEN
                              ngrav_fc_subs = ngrav_fc_subs + 1
                              n_gated_grav_outlets_fc(n) =
     $                        n_gated_grav_outlets_fc(n) + 1
                           ENDIF
                        ELSE
                           IF (ipurpse(n,i,ns).ne.1)
     $                     THEN
                              IF (ngrav_fc_subs .eq. 0) THEN
                                 n_gated_grav_outlets_fc(n) =
     $                           n_gated_grav_outlets_fc(n) + 1
                              ENDIF
                              ngrav_fc_subs = ngrav_fc_subs + 1
                           ENDIF
                        ENDIF
                     ENDIF
                  ENDIF
               ENDDO
            ENDIF
         ENDDO
         CALL match(upstrm_canal_name(n),1,5,cnm,nch+1,5,imatch)
         iup_canal_ws(n) = imatch
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF ( NOUT(N) .GT. 0 .OR. NSTRCTR(N) .GT. 0) THEN
            NPRSTRMN(N) = MAXSTR+NOUT(N)
            NOUTADD = NOUT(N)
            DO MM = MAXSTR+1,MAXSTR+NOUTADD
               STRNAME(MM) = weir_name(N,MM-MAXSTR)
            END DO
            DO MM = 1,NSTRCTR(N)
               TMPNAME(N,MM) = str_id(N,MM)
            END DO
            MAXSTR = MAXSTR + NOUT(N) + NSTRCTR(N)
C
```

```
         ENDIF
C------------------
C  SUM NUMBER OF STRUCTURES TO BE MANAGED IN WROUTE SUBROUTINE
C------------------
         IF (NSTRCTR(N) .GT. 0) THEN
            DO IS = 1,NSTRCTR(N)
               NSTR = NSTR + 1
               IF (IS .EQ. 1) ISTRMIN(N) = NSTR
            END DO
         ENDIF
C
         DO i = 1,nstrctr(n)
            IF (nprty(n,i) .EQ. 0) THEN
               strname(nprstrmn(n)+i) = tmpname(n,i)
            ELSE
               strname(nprstrmn(n)+i) = tmpname(n,nprty(n,i))
            ENDIF
         END DO
         CHDEP(N) = BEGSTG(N)
  190 CONTINUE
c
c  check if upstream canal is listed after downstream canal for water
c  supply purposes
c
         DO N = 1,NCH-1
           IF (upstrm_canal_name(n) .ne. 'NOCNL') THEN
             DO NCWS = N+1,NCH
                IF (cnm(NCWS) .eq. upstrm_canal_name(n)) THEN
                   WRITE(*,1050) cnm(n),cnm(ncws),cnm(n),cnm(ncws)
                   STOP
               ENDIF
             ENDDO
           ENDIF
         ENDDO
c
c  calculate canal water surface area in each grid cell
c
         NPTS_CN = 0
         DO N = 1,NCH
            DO IC = 1,NODCR(N)
               NPTS_CN = NPTS_CN + 1
               icell_no = NODECN(NPTS_CN)/10
               no_canl_in_cell(icell_no) = no_canl_in_cell(icell_no) + 1
               carea_in_cell(icell_no) = carea_in_cell(icell_no)
     &         + DISTNC(ICL(N,IC)) * WIDTH(N)
               rindiv_canl_area(icell_no,no_canl_in_cell(icell_no)) =
     &         DISTNC(ICL(N,IC)) * WIDTH(N)
            ENDDO
         ENDDO
         do node = 1,max_ncells
            carfact(node) = carea_in_cell(node)/GDAR
            DO itype = 1,num_use_type
               frac_irr_area(node,itype) = frac_irr_area(node,itype)
     &         / (1.0 - carfact(node))
            ENDDO
            frac_irr_area_t(node) = frac_irr_area_t(node)
     &         / (1.0 - carfact(node))
```

```
            IF (use_lec_et * lec_node_in_model(node) .NE. 0 .AND. ibsn
     $      (node) .NE. 40 .AND. frac_irr_area_t(node) .NE. 0.) THEN
               IF (erzd(node) .lt. 0.00001) THEN
                 whcap = 0.0
               ELSE
                 whcap = smmax(node) / ((GDAR/43560.)*erzd(node))
                 whcap = whcap / (1.0 - carfact(node))
                 IF (s(node) .LT. whcap) THEN
                    s(node) = whcap
                  ENDif
               ENDif
            ENDIF
         enddo
c
c  check if canal network for demand calculation is properly defined
c  in input
c
         DO N = 1,NCH
            ibranch_ws = 0
            DO NWS = N,NCH
               IF (upstrm_canal_name(nws) .EQ. cnm(n)) THEN
                  ibranch_ws = ibranch_ws + 1
               ENDIF
            ENDDO
            IF (NBRANCH(N) .EQ. 1) THEN
               IF (ibranch_ws .GT. 1) THEN
                   WRITE(*,5) ibranch_ws,cnm(n),NBRANCH(N)
                   STOP
               ENDIF
            ELSE
               IF (ibranch_ws .NE. NBRANCH(N)) THEN
                   WRITE(*,5) ibranch_ws,cnm(n),NBRANCH(N)
                   STOP
               ENDIF
            ENDIF
         ENDDO
c
c  input structures with operations triggered by locations other than
c  immediate upstream headwater or canal
c
      READ (11,*) nsfc_w_chg_trg
      DO kk = 1,nsfc_w_chg_trg
         READ (11,*) str_name_chg_loc_fc_ops,cnl_name_upstrm_fc_ops
     $   ,cnl_name_trg_ops,ixfc_trg_ops(kk,1),iyfc_trg_ops(kk,1)
     $   ,str_name_alt_loc_trg_ops,cnl_name_alt_trg_ops,ixfc_trg_ops
     $   (kk,2),iyfc_trg_ops(kk,2)
         igrid_loc_trg_ops(kk,1) = ixfc_trg_ops(kk,1)-minx(iyfc_trg_ops
     $   (kk,1))+ 1 + isum(iyfc_trg_ops(kk,1))
c
         IF (ixfc_trg_ops(kk,2) .gt. 0 .and. iyfc_trg_ops(kk,2) .gt. 0)
     $    THEN
            igrid_loc_trg_ops(kk,2)=ixfc_trg_ops(kk,2)-minx(iyfc_trg_ops
     $      (kk,2))+ 1 + isum(iyfc_trg_ops(kk,2))
         ELSE
            igrid_loc_trg_ops(kk,2)= -901
         ENDIF
c
```

```
      call match(cnl_name_trg_ops,1,5,cnm,nch+1,5,imatch_trg_ops)
      IF (imatch_trg_ops .eq. 0) THEN
         WRITE(*,6) cnl_name_trg_ops,str_name_chg_loc_fc_ops
         STOP
      ENDIF
      icanl_no_trg_ops(kk,1) = imatch_trg_ops
c
      call match(cnl_name_alt_trg_ops,1,5,cnm,nch+1,5,imatch)
      IF (imatch .eq. 0) THEN
         WRITE(*,665) cnl_name_alt_trg_ops,str_name_chg_loc_fc_ops
         STOP
      ENDIF
      icanl_no_trg_ops(kk,2) = imatch
      call match(cnl_name_upstrm_fc_ops,1,5,cnm,nch,5,imatch)
      IF (imatch .eq. 0) THEN
         WRITE(*,7) cnl_name_upstrm_fc_ops,str_name_chg_loc_fc_ops
         STOP
      ENDIF
      icnl_no_upstrm_trg_ops(kk) = imatch
c
      DO istruc = 1,nstrctr(icnl_no_upstrm_trg_ops(kk))
         outf_struc_names_ops(istruc) =tmpname(icnl_no_upstrm_trg_ops
     $      (kk),istruc)
      ENDDO
      call match(str_name_chg_loc_fc_ops,1,6,outf_struc_names_ops
     $   ,nstrctr(icnl_no_upstrm_trg_ops(kk)),6,imatch)
      IF (imatch .eq. 0) THEN
         WRITE(*,8) str_name_chg_loc_fc_ops
         STOP
      ENDIF
      istruc_no_chg_trg_ops(kk) = imatch
c
      call match(str_name_alt_loc_trg_ops,1,6,flnm,300,6,imatch)
      IF (str_name_alt_loc_trg_ops .ne. 'NOSTR ') THEN
        IF (imatch .eq. 0) THEN
          WRITE(*,667) str_name_alt_loc_trg_ops
          STOP
        ENDIF
        istruc_no_alt_trg_ops(kk) = imatch
      ELSE
       istruc_no_alt_trg_ops(kk) = -901
      ENDIF

c
      DO ll = 1,2
       IF (cnm(icanl_no_trg_ops(kk,ll)) .ne. 'NOCNL') THEN
         DO icnode = 1,nodcr(icanl_no_trg_ops(kk,ll))
            IF (ixfc_trg_ops(kk,ll).eq.xcn(icanl_no_trg_ops(kk,ll)
     $         ,icnode) .AND. iyfc_trg_ops(kk,ll).eq.ycn
     $         (icanl_no_trg_ops(kk,ll),icnode)) THEN
               node_pos_trg_ops(kk,ll) = icnode
               IF (icnl_no_upstrm_trg_ops(kk) .eq. icanl_no_trg_ops
     $            (kk,ll)) THEN
                  IF (node_pos_trg_ops(kk,ll) .lt. mid_point_canl(
     $                icnl_no_upstrm_trg_ops(kk))) THEN
                    itrg_loc_ops_cnl(kk,ll) = 1
                  ELSE
```

```
                      itrg_loc_ops_cnl(kk,ll) = 2
                    ENDIF
                  ELSE
                    itrg_loc_ops_cnl(kk,ll) = 3
                  ENDIF
               ENDIF
            ENDDO
            IF (node_pos_trg_ops(kk,ll) .eq. 0) THEN
               WRITE(*,9) ixfc_trg_ops(kk,ll),iyfc_trg_ops(kk,ll)
     $         ,cnl_name_trg_ops,str_name_chg_loc_fc_ops
               STOP
            ENDIF
          ELSE
           node_pos_trg_ops(kk,ll) = -901
           itrg_loc_ops_cnl(kk,ll) = -901
          ENDIF
         ENDDO
      ENDDO
c
c  input structures with single on/off downstream trigger (at any location)
c
      READ(11,*) no_struc_sing_on_off
      DO kk = 1,no_struc_sing_on_off
         READ (11,*) str_name_on_off,ups_canal_name_on_off
     $   ,canal_name_on_off,ixgr_on_off(kk)
     $   ,iygr_on_off(kk),trig_stg_on_off(kk),offset_on_off(kk)
         igrid_loc_on_off(kk) = ixgr_on_off(kk)-minx(iygr_on_off(kk))
     $   + 1 + isum(iygr_on_off(kk))
         call match(canal_name_on_off,1,5,cnm,nch+1,5,imatch)
         IF (imatch .eq. 0) THEN
            WRITE(*,17) canal_name_on_off,str_name_on_off
            STOP
         ENDIF
         icnl_no_on_off(kk) = imatch
c
         call match(ups_canal_name_on_off,1,5,cnm,nch+1,5,imatch)
         IF (imatch .eq. 0) THEN
            WRITE(*,177) ups_canal_name_on_off,str_name_on_off
            STOP
         ENDIF
         iups_cnl_no_on_off(kk) = imatch
c
         DO istruc = 1,nstrctr(iups_cnl_no_on_off(kk))
            outf_struc_names_on_off(istruc) =tmpname(iups_cnl_no_on_off
     $      (kk),istruc)
         ENDDO
         call match(str_name_on_off,1,6,outf_struc_names_on_off
     $   ,nstrctr(iups_cnl_no_on_off(kk)),6,imatch)
         IF (imatch .eq. 0) THEN
            WRITE(*,18) str_name_on_off
            STOP
         ENDIF
         istruc_no_on_off(kk) = imatch
c
         IF (cnm(icnl_no_on_off(kk)) .ne. 'NOCNL') THEN
            DO icnode = 1,nodcr(icnl_no_on_off(kk))
               IF (ixgr_on_off(kk) .eq. xcn(icnl_no_on_off(kk),icnode)
```

```
     $         .AND. iygr_on_off(kk) .eq. ycn(icnl_no_on_off(kk)
     $         ,icnode)) THEN
                  node_pos_on_off(kk) = icnode
               ENDIF
            ENDDO
            IF (node_pos_on_off(kk) .eq. 0) THEN
               WRITE(*,19) ixgr_on_off(kk),iygr_on_off(kk)
     $         ,canal_name_on_off,str_name_on_off
               STOP
            ENDIF
         ELSE
           node_pos_on_off(kk) = -901
         ENDIF
      ENDDO
c
c
c  input structures with time varying capacities(other than S12s,S343AB)
c
      READ (11,*) no_struc_tvary
      DO kk = 1,no_struc_tvary
         READ (11,*) struc_tvary_name
         DO n = 1,nch
            DO ii = 1,nstrctr(n)
               IF (flnm(istr(n,ii)) .EQ. struc_tvary_name) THEN
                  icanal_struc_vary = n
                  istruc_no_vary = ii
                  GO TO 200
               ENDIF
            END DO
         END DO
  200    READ (11,*) (frac_struc_cap(icanal_struc_vary,istruc_no_vary,
     $   im) ,im = 1,12)
      END DO
c
c   input canal names and minimum levels maintained during dry periods
c   when WCAs are below respective floor elevation
c
      READ (11,*) no_canl_min_mod_bel_flr,month_beg_min_mod_flr
     $ ,iday_beg_min_mod_flr,month_end_min_mod_flr
     $ ,iday_end_min_mod_flr
      DO ic = 1,no_canl_min_mod_bel_flr
          READ (11,*) canl_min_mod_name,(crmind_bflr(ic,i),i=1,2)
          call match (canl_min_mod_name,1,5,cnm,nch+1,5,imatch)
          icnl_min_mod_indx(ic) = imatch
      ENDDO
      DO ic = 1,no_canl_min_mod_bel_flr
          CRMIN_bel_flr(icnl_min_mod_indx(ic),1) = crmind_bflr(ic,1)
          CRMIN_bel_flr(icnl_min_mod_indx(ic),2) = crmind_bflr(ic,2)
      ENDDO
c
c   input seasonal flood control elevations for following structures
c   simulated as fixed crest weirs
c
      READ (11,*) nreg_seas
      DO i = 1,nreg_seas
         READ (11,'(A5,2x,A6,2F6.1)') canal_name_reg_seas
     $   ,struc_name_reg_seas,(reg(k),k=1,2)
```

```
         call match (canal_name_reg_seas,1,5,cnm,nch+1,5,imatch)
         icnl_no_reg_seas(i) = imatch
         nout_reg_seas = nout(icnl_no_reg_seas(i))
         do is = 1,nout(icnl_no_reg_seas(i))
            weir_name_reg_seas(is) = weir_name(icnl_no_reg_seas(i),is)
            weir_reg_seas(i,is,1) = reg(1)
            weir_reg_seas(i,is,2) = reg(2)
         enddo
         call match (struc_name_reg_seas,1,6,weir_name_reg_seas
     $    ,nout_reg_seas,6,imatch)
         iweir_no_reg_seas(i) = imatch
      ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      DO ia = 1,ntotres
         IF ( nasr_from_res(ia) .GT. 0) THEN
            DO j = 1,nasr_from_res(ia)
               MAXSTR = MAXSTR + 1
               STRNAME(MAXSTR) = asr_name(ia,j)
            END DO
         ENDIF
         IF (nasr_to_res(ia) .GT. 0) THEN
            DO j = 1,nasr_to_res(ia)
               MAXSTR = MAXSTR + 1
               STRNAME(MAXSTR) = asr_name_rec(ia,j)
            END DO
         ENDIF
         DO i = 1,noutstr(ia)
            IF (res_out_type(ia,i) .NE. 'WSPLY '.AND.
     $       res_out_type(ia,i) .NE. 'SEEPG') THEN
               MAXSTR = MAXSTR + 1
               STRNAME(MAXSTR) = res_out_strname(ia,i)
            ENDIF
         END DO
      END DO
c     PRINT *,MAXSTR,'  JUST BEF PASSIVE WEIR NAMES'
C
      DO i = 1,no_of_wiers
         ndrctn = 0
         DO k = 1,5
            IF (wier_orient(i,k) .NE. 'O') THEN
               ndrctn = ndrctn + 1
               MAXSTR = MAXSTR + 1
               STRNAME(MAXSTR) = weir_name_wca(i,ndrctn)
            ENDIF
         END DO
      END DO
      READ (11,*) no_if_conditions
      DO icond = 1,no_if_conditions
c  ----------------------------------------
c     icond = 1 - unconditionally output,relative to current base
c     icond = 2 - if number of stas(nmarea) greater than zero
c     icond = 3 - if number of stas(nmarea) .GT. 0 or makeup rule
c                  is in effect
c     icond = 4 - if Miami Canal Basin runoff and NNR Canal basin
c                 runoff in EAA is diverted to STA34
c     icond = 5 - if L-8 basin current plumbing system and operation
c                  is simulated (opt_l8_prop = 0)
```

```
c     icond = 6 - if proposed L-8 basin plumbing system and operation
c                  is simulated (opt_l8_prop = 1)
c     icond = 7 - if Rainfall Plan or NSM stage targets used in ENP
c                 (type_flow_across_ttrail .NE. 'MINDEL')
c     icond = 8 - if North storage is simulated for LOK
c     icond = 9 - if seminole demands in Big Cypress Basin are simulated
c                 (isem_flg = 1)
c     icond = 10 - if LOK is split into sections
c
         READ (11,'(i5,2x,20(A6,1x),11(/7X,20(A6,1x)))') NADDSTR,
     $(STRNAME(I),I=MAXSTR+ 1,MAXSTR+NADDSTR)
         n_str_if_cond(icond) = NADDSTR

         MAXSTR = MAXSTR + NADDSTR
c        print *,maxstr,'  KKKKK', NADDSTR
c        print *,n_str_if_cond(icond)
      END DO
c
c  READ if nmaws > 0
      n_sta_inf_dss = 0
      IF (nstarff .GT. 0) THEN
         READ (11,'(I5,2x,10(A6,1x))') n_sta_inf_dss,(STRNAME(I),
     $   I=MAXSTR+ 1,MAXSTR+NSTARFF)
         MAXSTR = MAXSTR + NSTARFF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      n_str_add = 0
      do kkk = 1,max_n_res
        ieaarsc(kkk) = 0
        ieaarreg(kkk) = 0
      enddo
      naddstr_asr_eaa = 0
      IF (iasr_opt_eaa(is8bsn - 6).EQ. 1.OR. iasr_opt_eaa(is7bsn - 6)
     +  .EQ. 1 .OR. iasr_opt_eaa(is5absn-6) .EQ. 1 .OR. nasr_to_eaa
     +  (is8bsn - 6) .GT. 0 .OR. nasr_to_eaa(is7bsn - 6) .GT. 0 .OR.
     +  nasr_to_eaa(is5absn - 6) .GT. 0) THEN
         READ (11,'(I5,2x,10(A6,1x))') NADDSTR_ASR_EAA,(STRNAME(I)
     +   ,I=MAXSTR+ 1, MAXSTR+NADDSTR_ASR_EAA)
         MAXSTR = MAXSTR + NADDSTR_ASR_EAA
      ENDIF
      DO ib = 1,neaabsn
         IF (ib .NE. 2) THEN
            IF (iasr_opt_eaa(ib).EQ. 1 .OR. ires_opt_eaa(ib).EQ. 1)THEN
                n_str_add = n_str_add + 1
                ieaarsc(ieaa_res_asr(ib)) = ieaarsc(ieaa_res_asr(ib))
     $          + 1
                STRNAME(MAXSTR+n_str_add) = res_struc_inflow_name
     $          (ieaa_res_asr(ib),ieaarsc(ieaa_res_asr(ib)))
            ENDIF
            DO kk = 1,no_of_res_reg_frm_lok(ib)
               n_str_add = n_str_add + 1
               ieaarreg(ieaa_res_asr_reg(ib,kk))
     $         = ieaarreg(ieaa_res_asr_reg(ib,kk)) + 1
               STRNAME(MAXSTR+n_str_add) = res_struc_inflow_name
     $         (ieaa_res_asr_reg(ib,kk),ieaarreg(ieaa_res_asr_reg(ib,kk)
     $         ))
            END DO
```

```
         ELSE
            IF (iasr_opt_eaa(ib) .EQ. 1 .OR. ires_opt_eaa(ib) .EQ. 1
     $      .OR. ires_opt_hill .EQ. 1) THEN
               n_str_add = n_str_add + 1
               ieaarsc(ieaa_res_asr(ib)) = ieaarsc(ieaa_res_asr(ib))
     $         + 1
               STRNAME(MAXSTR+n_str_add) = res_struc_inflow_name
     $         (ieaa_res_asr(ib),ieaarsc(ieaa_res_asr(ib)))
            ENDIF
            DO kk = 1,no_of_res_reg_frm_lok(ib)
               n_str_add = n_str_add + 1
               ieaarreg(ieaa_res_asr_reg(ib,kk))
     $         = ieaarreg(ieaa_res_asr_reg(ib,kk)) + 1
               STRNAME(MAXSTR+n_str_add) = res_struc_inflow_name
     $         (ieaa_res_asr_reg(ib,kk),ieaarreg(ieaa_res_asr_reg(ib,kk)
     $         ))
            END DO
         ENDIF
      END DO
      IF (n_str_add .GT. 0) MAXSTR = MAXSTR + n_str_add
c
      n_str_add = 0
      DO irs = 1,nres_ws
         DO js = 1,noutlets(irs)
            IF (idshg_opt(ires_no(irs),js) .EQ. 1) THEN
               n_str_add = n_str_add + 1
               STRNAME(MAXSTR+n_str_add) = res_out_strname
     $         (ires_no(irs),istrctr_ws(ires_no(irs),js))
            ENDIF
         END DO
      END DO
      IF (n_str_add .GT. 0) MAXSTR = MAXSTR + n_str_add
c
      IF (ires_opt_eaa(is8bsn-6).EQ. 1.OR. ires_opt_eaa(is7bsn-6) .EQ.
     $ 1 .OR. ires_opt_eaa(is5absn-6) .EQ. 1 .OR. ires_opt_hill .EQ. 1)
     $THEN
         READ (11,'(i5,2x,20(A6,1x),10(/7X,20(A6,1x)))') NADDSTR,
     $(STRNAME(I),I=MAXSTR+ 1,MAXSTR+NADDSTR)
         MAXSTR = MAXSTR + NADDSTR
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      n_str_add = 0
      DO irs = 1,nres_ws
         IF (nasr_from_res(ires_no(irs)) .GT. 0) THEN
            DO is = 1,noutlets(irs)
               js = istrctr_ws(ires_no(irs),is)
               IF (idshg_opt(ires_no(irs),is) .NE. 1) THEN
                  IF (inodopt(ires_no(irs),js,3) .EQ. 0) THEN
                     icnl_no = ircnl(ires_no(irs),js,1)
                     char_id_cnl_ws_resasr = cnm(icnl_no)
                     IF (char_id_cnl_ws_resasr(1:3) .ne. 'LWD') THEN
                       IF (nasr_to_cnl(icnl_no) .GT. 0) THEN
                         n_str_add = n_str_add + 1
                       ENDIF
                     ENDIF
                  ENDIF
               ENDIF
```

```
            END DO
         ENDIF
      END DO
      IF (n_str_add .GT. 0) THEN
         READ (11,'(7x,20(A6,1x),10(/7X,20(A6,1x)))')
     $(STRNAME(I),I=MAXSTR+ 1,MAXSTR+n_str_add)
         MAXSTR = MAXSTR + n_str_add
      ENDIF
C
      IF (LP(1)) READ(55,'(1X,F6.2)') stagelo
      IF (lp(1)) THEN
         IF (runmode .EQ. 'CALIB') THEN
            READ(55,*) no_canals
            IF (nch .EQ. no_canals) READ(55,'(15F6.2)') (CHDEP(IN) ,IN
     $      =1,no_canals)
         ELSE
            READ(55,'(15F6.2)') (CHDEP(IN),IN=1,NCH)
         ENDIF
      ENDIF
      IF (initialize_annually_opt .EQ. 1) THEN
         DO n = 1,nch
            chdep0(n) = chdep(n)
         END DO
         stagelo0 = stagelo
      ENDIF
c
C--------------------------
C   READ LEVEE SPECIFICATIONS FOR CALCULATION OF
C     LEVEE SEEPAGE
C--------------------------
      READ(104,'(i5,2x,20(A5,1x))') nlevs,(rlist_lvname(i),i=1,nlevs)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      L = 0
      DO 260 K = 1,NLEVS
         READ  (104,'(5x,A5)') LVNAME(K)
         CALL match (lvname(k),1,5,rlist_lvname,nlevs,5,imatch)
         IF (imatch .EQ. 0) THEN
            WRITE (*,230) lvname(k)
            STOP
         ENDIF
         lev_no(K) = imatch
         KK = lev_no(K)
         READ (104,'(I5,5F7.2)') LVNODES(KK),SRATE(KK,1),SRATE(KK,2) ,
     $   SRATE(KK,3),srate_frac(KK),rate_limit(KK)
         DO 250 IL = 1,LVNODES(KK)
            L = L + 1
            READ (104,240) (LVSP(L,IK),IK = 1,3),
     $      borrow_cnl_name_upstrm ,borrow_cnl_name_dnstrm,(LVSP(L,IK),
     $      IK = 6,8)
            IF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 32)
     $      rlv_seep_to_canl_c11b = borrow_cnl_name_dnstrm
            READ (104,*) lvseep_divers_str_name(L),lvseep_pump_cap(l) ,
     $      opt_dest_lvseep(l),(frac_seep_divers(l,i),i=1,2)
            IF (opt_dest_lvseep(l) .EQ. 1) THEN
               READ (104,*) no_dest_lvseep(l),(icol_dest_lvseep(i) ,
     $         irow_dest_lvseep(i),i=1,no_dest_lvseep(l))
               DO i = 1,no_dest_lvseep(l)
```

```
                  idest_cell_loc_lvseep(l,i) = icol_dest_lvseep(i) -
     $            minx(irow_dest_lvseep(i)) +  1+ isum(irow_dest_lvseep
     $            (i))
               END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            ELSE
               READ (104,'(i3,2x,5(A5,2x))') no_dest_lvseep(l) ,
     $         (dest_canal_name_lvseep(i),i=1,no_dest_lvseep(l))
               DO i = 1,no_dest_lvseep(l)
                  CALL match (dest_canal_name_lvseep(i),1,5,cnm,nch+ 1,
     $            5,imatch)
                  idest_canal_lvseep_index(l,i) = imatch
               END DO
            ENDIF
            IF (nint(lvseep_pump_cap(l)) .GT. 0) THEN
               maxstr = maxstr + 1
               strname(maxstr) = lvseep_divers_str_name(L)
            ENDIF
            CALL match(borrow_cnl_name_dnstrm,1,5,cnm,nch+1, 5,
     $      imatch_cnl)
            IF (imatch_cnl .eq. 0) THEN
                WRITE(*,960) borrow_cnl_name_dnstrm,lvname(k)
                STOP
            ENDIF
            LVSP(L,5) = imatch_cnl
            CALL match(borrow_cnl_name_upstrm,1,5,cnm,nch+1, 5,
     $      imatch_cnl)
            IF (imatch_cnl .eq. 0) THEN
                WRITE(*,970) borrow_cnl_name_upstrm,lvname(k)
                STOP
            ENDIF
            LVSP(L,4) = imatch_cnl
c
c  diagnostic-check if input is correct
c  STOP execution of program if NOT correct
c
            IF (LVSP(L,4) .le. nch) THEN
               icorrect_locu = 0
               DO IPOS = 1,NODCR(LVSP(L,4))
                  IF (LVSP(L,1) .EQ. XCN(LVSP(L,4),ipos) .AND. LVSP(L,2)
     $              .EQ. ycn(LVSP(L,4),ipos)) THEN
                     icorrect_locu = 1
                  ENDIF
               ENDDO
               IF (icorrect_locu .eq. 0) THEN
                   WRITE(*,1111) LVSP(L,1),LVSP(L,2),
     $              borrow_cnl_name_upstrm
 1111              FORMAT(2(/),'GRID LOCATION,COL',2x,I3,2x,'ROW',2x,I3
     $,2x,'IS NOT LOCATED ALONG UPSTREAM CANAL,',2x,A6,3x,'FOR LEVEE SEE
     $PAGE. CHECK GRID LOCATION OR CANAL NAME IN levee_spg_input.dat FIL
     $E')
                   STOP
               ENDIF
            ENDIF

            IF (LVSP(L,4) .le. nch) THEN
               DO IPOS = 1,NODCR(LVSP(L,4))
```

```
                  IF (LVSP(L,1) .EQ. XCN(LVSP(L,4),ipos) .AND. LVSP(L,2)
     $            .EQ. ycn(LVSP(L,4),ipos)) THEN
                     ICNL_NODE_POS(L,1) = ipos
                  ENDIF
               END DO
               if (ICNL_NODE_POS(L,1).le.mid_point_canl(LVSP(L,4))) THEN
                   ilvflw_for_slope(l,1) = 2
               else
                   ilvflw_for_slope(l,1) = 3
               endif
            ENDIF
c
c  diagnostic-check if input is correct
c  STOP execution of program if NOT correct
c
            IF (LVSP(L,5) .le. nch) THEN
               icorrect_locd = 0
               DO IPOS = 1,NODCR(LVSP(L,5))
                  IF (LVSP(L,6) .EQ. XCN(LVSP(L,5),ipos) .AND. LVSP(L,7)
     $              .EQ. ycn(LVSP(L,5),ipos)) THEN
                     icorrect_locd = 1
                  ENDIF
               ENDDO
               IF (icorrect_locd .eq. 0) THEN
                   WRITE(*,1112) LVSP(L,6),LVSP(L,7),
     $              borrow_cnl_name_dnstrm
 1112              FORMAT(2(/),'GRID LOCATION,COL',2x,I3,2x,'ROW',2x,I3
     $,2x,'IS NOT LOCATED ALONG DNSTREAM CANAL,',2x,A6,3x,'FOR LEVEE SEE
     $PAGE. CHECK GRID LOCATION OR CANAL NAME IN levee_spg_input.dat FIL
     $E')
                   STOP
               ENDIF
            ENDIF
            IF (LVSP(L,5) .le. nch) THEN
               DO IPOS = 1,NODCR(LVSP(L,5))
                  IF (LVSP(L,6) .EQ. XCN(LVSP(L,5),ipos) .AND. LVSP(L,7)
     $            .EQ. ycn(LVSP(L,5),ipos)) THEN
                     ICNL_NODE_POS(L,2) = ipos
                  ENDIF
               END DO
               if (ICNL_NODE_POS(L,2).le.mid_point_canl(LVSP(L,5))) THEN
                   ilvflw_for_slope(l,2) = 2
               else
                   ilvflw_for_slope(l,2) = 3
               endif
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0--
  250    CONTINUE
  260 CONTINUE
c
cbw 1-17-01 commented out this:   "define index for L-30 , need later"
      CALL match('L30  ',1,5,rlist_lvname,nlevs,5,ilv30)
C
C-----------------------------------
C  READ CURTAIN WALL LOCATIONS
C-----------------------------------
  270 READ (104,'(2I5)',END = 280) ICOL_CW,IROW_CW
```

```
      NODE_CW = ICOL_CW - MINX(IROW_CW) +  1+ ISUM(IROW_CW)
      READ (104,'(3X,5(A1,1X))') (BORIENT(NODE_CW,I),I=1,4)
      READ (104,'(2x,4(2i1,1X))') ((icurtw_opt(NODE_CW,I,J),I=1,2) ,J=1,
     $4)
      ICURTAIN(NODE_CW) = 1
      GO TO 270
  280 LMAX=L
      WRITE(32,290)LMAX
      WRITE(32,300)(LVSP(L,1),LVSP(L,2),L=1,LMAX)
      WRITE(32,310)(LVSP(L,6),LVSP(L,7),L=1,LMAX)
CLMB<<
C-------------------------
C-----------------
      PRINT 320, NCH, NSTEP ,(cnm(mtrc(i)),i=1,nmc)
      CLOSE (11)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      CLOSE (23)
      CLOSE (104)
c
c   input data for pre- storm drawdown
c
      READ (136,*) ndays_prestorm_drawdown
c
      IF (ndays_prestorm_drawdown .GT. 0) THEN
c
c   for canals/structures
c
         idrwn = 0
  330    idrwn = idrwn + 1
         READ (136,*,end = 340) canal_name_drawdown ,struc_name_drawdown
     $   (idrwn),rmxstg_drawdown(idrwn) ,rmnstg_drawdown(idrwn) ,
     $   (tw_max_struc_inflw_drwdwn(idrwn,j),j=1,3)
c
         CALL match (canal_name_drawdown,1,5,cnm,nch+1, 5,imatch_drwdwn)
         IF (imatch_drwdwn .EQ. 0) THEN
             WRITE(*,980) canal_name_drawdown
             STOP
         ENDIF
c
         icanal_no_drawdown(idrwn) = imatch_drwdwn
         IF (struc_name_drawdown(idrwn) .NE. 'WIER  ') THEN
            DO istruc = 1,nstrctr(icanal_no_drawdown(idrwn))
               outf_struc_names(istruc) = tmpname(icanal_no_drawdown
     $         (idrwn),istruc)
            END DO
            CALL match(struc_name_drawdown(idrwn),1,6,outf_struc_names,
     $      nstrctr(icanal_no_drawdown(idrwn)),6,imatch_struc_drwdwn)
            IF (imatch_struc_drwdwn .eq. 0) THEN
               WRITE(*,990) struc_name_drawdown(idrwn)
               STOP
            ENDIF
            istruc_no_drawdown(idrwn) = imatch_struc_drwdwn
         ENDIF
         GO TO 330
  340    CONTINUE
         idrwn = idrwn - 1
c
```

```
c  for reservoirs/outlet structures
c
         idrwnres = 0
  342    idrwnres = idrwnres + 1
         READ (147,344,end = 345) res_name_drawdown,
     $   res_struc_name_drwdown(idrwnres),CRSTELEV_drwdwn(idrwnres),
     $   stgmx_drwdwn(idrwnres)
c
         call match (res_name_drawdown,1,6,resname,ntotres,6
     $   ,imatch_res_drwdwn)
         IF (imatch_res_drwdwn .eq. 0) THEN
             WRITE(*,1000) res_name_drawdown
             STOP
         ENDIF
         ires_no_drawdown(idrwnres) = imatch_res_drwdwn
         do istruc = 1,noutstr(ires_no_drawdown(idrwnres))
           outf_struc_names_res(istruc)=res_out_strname(ires_no_drawdown
     $     (idrwnres),istruc)
         enddo
         call match (res_struc_name_drwdown(idrwnres),1,6,
     $   outf_struc_names_res,noutstr(ires_no_drawdown(idrwnres)),
     $   6,imatch_struc_res_drwdwn)
         IF (imatch_struc_res_drwdwn .eq. 0) THEN
             WRITE(*,1010) res_struc_name_drwdown(idrwnres)
             STOP
         ENDIF
         istruc_no_res_drwdwn(idrwnres) = imatch_struc_res_drwdwn
         go to 342
  345    continue
         idrwnres = idrwnres - 1
c
         DO index = 1,max_n_days_sim
            nstorm_day(index) = 0
         END DO
c
         ifyr1 = ifyr
         ibm1 = ibm
         julian_day_beg = julian_ymd(ifyr1,ibm1,1)
         nstorm = 0
  350    nstorm = nstorm + 1
         READ (137,*,end = 360) month_storm(nstorm),iday_storm(nstorm),
     $   iyear_storm(nstorm)
         GO TO 350
c
c   determine days of storm drawdown
c
  360    DO istorm = 1,nstorm-1
            julian_day_storm = julian_ymd(iyear_storm(istorm) ,
     $      month_storm(istorm),iday_storm(istorm))
            index_day_storm = julian_day_storm - julian_day_beg + 1
            nstorm_day(index_day_storm) = 1
            DO iduration = 1,ndays_prestorm_drawdown-1
               nstorm_day(index_day_storm+iduration) = 1
            END DO
         END DO
      ENDIF
c
```

```
c  for canals with dual operations as a function of discharges elsewhere
c
         READ (157,*) no_of_flow_cond
         do no = 1,no_of_flow_cond
           READ (157,*) ndual_canal_ops(no)
           DO no1 = 1,ndual_canal_ops(no)
              READ (157,*) canal_name_dual_ops,struc_name_dual_ops
     $        (no1,no),rmxstg_dual_ops(no1,no),rmnstg_dual_ops(no1,no),
     $        canal_dest_name_dual_ops,ndual_col_dest(no1,no),
     $        ndual_row_dest(no1,no)
              call match (canal_name_dual_ops,1,5,cnm,nch,5
     $        ,imatch_dualops)
              IF (imatch_dualops .eq. 0) THEN
                 WRITE(*,1020) canal_name_dual_ops
                 STOP
              ENDIF
              icanal_no_dualops(no1,no) = imatch_dualops
              call match ( canal_dest_name_dual_ops,1,5,cnm,nch+1,5
     $        ,imatch_dest)
              IF (imatch_dest .eq. 0) THEN
                 WRITE(*,1030) canal_dest_name_dual_ops
                 STOP
              ENDIF
              ndual_canl_dest_indx(no1,no) = imatch_dest
              if (imatch_dest .gt. nch) then
                 igrid_loc_dest_dualops(no1,no) = ndual_col_dest(no1,no)
     $           - minx(ndual_row_dest(no1,no)) + 1
     $           + isum(ndual_row_dest(no1,no))
              endif
c
              IF (struc_name_dual_ops(no1,no) .NE. 'WIER  ') THEN
                DO istruc = 1,nstrctr(icanal_no_dualops(no1,no))
                  outf_struc_names(istruc) = tmpname(icanal_no_dualops
     $           (no1,no),istruc)
                END DO
                CALL match(struc_name_dual_ops(no1,no),1,6,
     $           outf_struc_names,nstrctr(icanal_no_dualops(no1,no)),6,
     $           imatch_struc_dualops)
                IF (imatch_struc_dualops .eq. 0) THEN
                   WRITE(*,1040) struc_name_dual_ops(no1,no)
                   STOP
                ENDIF
                istruc_no_dualops(no1,no) = imatch_struc_dualops
                rmnstg_dual_ops0(icanal_no_dualops(no1,no)
     $           ,istruc_no_dualops(no1,no)) = rmnstg_dual_ops(no1,no)
              ENDIF
           ENDDO
         enddo
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  GET THE KNOWN FLOW POINTS SPECIFICATIONS AND WRITE THEM INTO
C       THE CNWORD ARRAY FOR USE IN SUBROUTINE KNFLOWS.
C-----------------
C     OPEN(22,FILE = 'kflpts2')
C-----------------
      PRINT 370
      II = 0
```

```
      NAGPTS = 0
      DO 630 I = 1,ISTR_COUNT
         II = II + 1
         READ (22,'(A6,2x,6i5)') FNM(i),iroute_opt(i),
     $   no_of_down_strm_loc(i),COLPOS_KU(i),ROWPOS_KU(i),
     $   COLPOS_KD(i),ROWPOS_KD(i)
         KFL(II,2) = no_of_down_strm_loc(i)
         KFL(II,1) = iroute_opt(i)
         IF (iroute_opt(i) .EQ. 0) THEN
            READ (22,*) canl_id,canl_id1,canl_id2,canl_id3
            CALL match (canl_id,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID
               STOP
            ENDIF
            KFL(II,3) = imatch
            CALL match (canl_id1,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID1
               STOP
            ENDIF
            KFL(II,4) = imatch
            CALL match (canl_id2,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID2
               STOP
            ENDIF
            KFL(II,5) = imatch
            CALL match (canl_id3,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID3
               STOP
            ENDIF
            KFL(II,6) = imatch
            PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,6)
         ELSE
            GO TO (380,390,420,430,450,470,490,510) iroute_opt(i)
380         READ (22,*) (KFL(II,JJ), JJ = 3,4)
            PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,4)
            GO TO 520
390         READ (22,*) CANL_ID
            CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID
               STOP
            ENDIF
            KFL(II,3) = imatch
            N = KFL(II,3)
            DO ipos = 1,nodcr(N)
               IF (COLPOS_KD(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KD
     $         (I) .EQ. YCN(N,IPOS)) THEN
                  NODEPOS_KD = IPOS
                  IF (NODEPOS_KD .eq. 1) THEN
                     isdnpflw_for_slope_kn(i) = 1
                  ELSEIF (NODEPOS_KD .le. mid_point_canl(N)
     $            .AND. NODEPOS_kd .ne. 1) THEN
                     isdnpflw_for_slope_kn(i) = 2
```

```
                ELSEIF (NODEPOS_KD .gt. mid_point_canl(N))
   $            THEN
                   isdnpflw_for_slope_kn(i) = 3
                ENDIF
                GO TO 395
             ENDIF
          END DO
395       PRINT 410,FNM(i),iroute_opt(i),KFL(II,2),CANL_ID
          GO TO 520
420       READ (22,*) (KFL(II,JJ),JJ = 3,6)
          PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,6)
          GO TO 520
430       READ (22,*) (KFL(II,JJ),JJ=3,4),CANL_ID
          CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
          IF (imatch .EQ. 0) THEN
             WRITE(*,400) CANL_ID
             STOP
          ENDIF
          KFL(II,5) = imatch
          N = KFL(II,5)
          DO ipos = 1,nodcr(N)
             IF (COLPOS_KD(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KD
   $         (I) .EQ. YCN(N,IPOS)) THEN
                NODEPOS_KD = IPOS
                IF (NODEPOS_KD .eq. 1) THEN
                   isdnpflw_for_slope_kn(i) = 1
                ELSEIF (NODEPOS_KD .le. mid_point_canl(N)
   $            .AND. NODEPOS_kd .ne. 1) THEN
                   isdnpflw_for_slope_kn(i) = 2
                ELSEIF (NODEPOS_KD .gt. mid_point_canl(N))
   $            THEN
                   isdnpflw_for_slope_kn(i) = 3
                ENDIF
                GO TO 445
             ENDIF
          END DO
445       PRINT 440,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,4),CANL_ID
          GO TO 520
450       READ (22,*) CANL_ID,(KFL(II,JJ),JJ=4,5)
          CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
          IF (imatch .EQ. 0) THEN
             WRITE(*,400) CANL_ID
             STOP
          ENDIF
          KFL(II,3) = imatch
          N = KFL(II,3)
          DO ipos = 1,nodcr(N)
             IF (COLPOS_KU(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KU
   $         (I) .EQ. YCN(N,IPOS)) THEN
                NODEPOS_KU = IPOS
                IF (NODEPOS_KU .eq. NODCR(N) .OR. fnm(i)
   $            .EQ. 'S12A  ' .OR. fnm(i) .EQ. 'S12B  ' .OR.
   $            fnm(i) .EQ. 'S12C  ' .OR. fnm(i) .EQ. 'S12D  '
   $            .OR. fnm(i) .EQ. 'S333  ') THEN
                   isuppflw_for_slope_kn(i) = 1
                ELSEIF (NODEPOS_KU .le. mid_point_canl(N))
   $            THEN
```

```
                  isuppflw_for_slope_kn(i) = 2
               ELSEIF (NODEPOS_KU .gt. mid_point_canl(N)
   $             .AND. NODEPOS_KU .ne. NODCR(N)) THEN
                  isuppflw_for_slope_kn(i) = 3
               ENDIF
               GO TO 465
            ENDIF
         END DO
465      PRINT 460,FNM(i),iroute_opt(i),kfl(ii,2),canl_id,(KFL(II,JJ)
   $     ,JJ=4,5)
         GO TO 520
470      READ (22,*) CANL_ID,CANL_ID2
         CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
         IF (imatch .EQ. 0) THEN
            WRITE(*,400) CANL_ID
            STOP
         ENDIF
         KFL(II,3) = imatch
         N = KFL(II,3)
         DO ipos = 1,nodcr(N)
            IF (COLPOS_KU(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KU
   $        (I) .EQ. YCN(N,IPOS)) THEN
               NODEPOS_KU = IPOS
               IF (NODEPOS_KU .eq. NODCR(N) .OR. fnm(i)
   $           .EQ. 'S12A  ' .OR. fnm(i) .EQ. 'S12B  ' .OR.
   $           fnm(i) .EQ. 'S12C  ' .OR. fnm(i) .EQ. 'S12D  '
   $           .OR. fnm(i) .EQ. 'S333  ') THEN
                  isuppflw_for_slope_kn(i) = 1
               ELSEIF (NODEPOS_KU .le. mid_point_canl(N))
   $           THEN
                  isuppflw_for_slope_kn(i) = 2
               ELSEIF (NODEPOS_KU .gt. mid_point_canl(N)
   $             .AND. NODEPOS_KU .ne. NODCR(N)) THEN
                  isuppflw_for_slope_kn(i) = 3
               ENDIF
               GO TO 475
            ENDIF
         END DO
475      CALL match (CANL_ID2,1,5,cnm,nch+1,5,imatch)
         IF (imatch .EQ. 0) THEN
            WRITE(*,400) CANL_ID2
            STOP
         ENDIF
         KFL(II,4) = imatch
         N = KFL(II,4)
         DO ipos = 1,nodcr(N)
            IF (COLPOS_KD(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KD
   $        (I) .EQ. YCN(N,IPOS)) THEN
               NODEPOS_KD = IPOS
               IF (NODEPOS_KD .eq. 1) THEN
                  isdnpflw_for_slope_kn(i) = 1
               ELSEIF (NODEPOS_KD .le. mid_point_canl(N)
   $           .AND. NODEPOS_kd .ne. 1) THEN
                  isdnpflw_for_slope_kn(i) = 2
               ELSEIF (NODEPOS_KD .gt. mid_point_canl(N))
   $           THEN
                  isdnpflw_for_slope_kn(i) = 3
```

```
                  ENDIF
                  GO TO 485
               ENDIF
            END DO
  485       PRINT 480,FNM(i),iroute_opt(i),kfl(ii,2),CANL_ID,CANL_ID2
            GO TO 520
  490       READ (22,*) (KFL(II,JJ),JJ=3,3),CANL_ID,KFL(II,5)
            CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,400) CANL_ID
               STOP
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            KFL(II,4) = imatch
            N = KFL(II,4)
            DO ipos = 1,nodcr(N)
               IF (COLPOS_KD(I) .EQ. XCN(N,IPOS) .AND. ROWPOS_KD
     $         (I) .EQ. YCN(N,IPOS)) THEN
                  NODEPOS_KD = IPOS
                  IF (NODEPOS_KD .eq. 1) THEN
                     isdnpflw_for_slope_kn(i) = 1
                  ELSEIF (NODEPOS_KD .le. mid_point_canl(N)
     $            .AND. NODEPOS_KD .ne. 1) THEN
                     isdnpflw_for_slope_kn(i) = 2
                  ELSEIF (NODEPOS_KD .gt. mid_point_canl(N))
     $            THEN
                     isdnpflw_for_slope_kn(i) = 3
                  ENDIF
                  GO TO 505
               ENDIF
            END DO
  505       PRINT 500,FNM(i),iroute_opt(i),(kfl(ii,jj),jj=2,3),canl_id
            GO TO 520
  510       READ(22,*) (KFL(II,JJ),JJ=3,6)
            PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,5)
  520       IF (((KFL(II,3) .GE. 7 .AND. KFL(II,3) .LE.10) .OR. KFL
     $      (II,3) .EQ. 40) .AND. (KFL(II,1) .GE. 7 .AND. KFL(II,1) .LE.
     $      8)) THEN
               NAGPTS = NAGPTS + 1
               IAGFLO(NAGPTS,2) = KFL(II,3)
            ENDIF
            IF (no_of_down_strm_loc(i) .LT. 2) GO TO 630
            DO J = 2,no_of_down_strm_loc(i)
               II = II + 1
               kfl(ii,2) = no_of_down_strm_loc(i)
               kfl(ii,1) = iroute_opt(i)
               GO TO (530,540,550,560,570,580,590,600) iroute_opt(i)
  530          READ (22,*) (KFL(II,JJ), JJ = 3,4)
               PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,4)
               GO TO 610
  540          READ (22,*) CANL_ID
               CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,400) CANL_ID
                  STOP
               ENDIF
               KFL(II,3) = imatch
```

```
              PRINT 410,FNM(i),iroute_opt(i),KFL(II,2),CANL_ID
              GO TO 610
550           READ (22,*) (KFL(II,JJ),JJ = 3,6)
              PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,6)
              GO TO 610
560           READ (22,*) (KFL(II,JJ),JJ=3,4),CANL_ID
              CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
              IF (imatch .EQ. 0) THEN
                 WRITE(*,400) CANL_ID
                 STOP
              ENDIF
              KFL(II,5) = imatch
              PRINT 440,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,4)
   $          ,CANL_ID
              GO TO 610
570           READ (22,*) CANL_ID,(KFL(II,JJ),JJ=4,5)
              CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
              IF (imatch .EQ. 0) THEN
                 WRITE(*,400) CANL_ID
                 STOP
              ENDIF
              KFL(II,3) = imatch
              PRINT 460,FNM(i),iroute_opt(i),kfl(ii,2),canl_id,(KFL
   $          (II,JJ),JJ=4,5)
              GO TO 610
580           READ (22,*) CANL_ID,CANL_ID2
              CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
              IF (imatch .EQ. 0) THEN
                 WRITE(*,400) CANL_ID
                 STOP
              ENDIF
              KFL(II,3) = imatch
              CALL match (CANL_ID2,1,5,cnm,nch+1,5,imatch)
              IF (imatch .EQ. 0) THEN
                 WRITE(*,400) CANL_ID2
                 STOP
              ENDIF
              KFL(II,4) = imatch
              PRINT 480,FNM(i),iroute_opt(i),kfl(ii,2),CANL_ID,CANL_ID2
              GO TO 610
590           READ (22,*) (KFL(II,JJ),JJ=3,3),CANL_ID,KFL(II,5)
              CALL match (CANL_ID,1,5,cnm,nch+1,5,imatch)
              IF (imatch .EQ. 0) THEN
                 WRITE(*,400) CANL_ID
                 STOP
              ENDIF
              KFL(II,4) = imatch
              PRINT 500,FNM(i),iroute_opt(i),(kfl(ii,jj),jj=2,3)
   $          ,canl_id
              GO TO 610
600           READ(22,*) (KFL(II,JJ),JJ=3,6)
              PRINT 620,FNM(i),iroute_opt(i),(KFL(II,JJ),JJ=2,5)
610           IF (((KFL(II,3) .GE. 7 .AND. KFL(II,3) .LE.10) .OR. KFL
   $          (II,3) .EQ. 40) .AND. (KFL(II,1) .GE. 7 .AND. KFL(II,1)
   $          .LE. 8)) THEN
                 NAGPTS = NAGPTS + 1
                 IAGFLO(NAGPTS,2) = KFL(II,3)
```

```
               ENDIF
            END DO
         ENDIF
  630 CONTINUE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (ISTAOPT(10) .EQ. 1) THEN
         NAGPTS = NAGPTS + 1
         IAGFLO(NAGPTS,2) = ISUGBSN
      ENDIF
C-----------------
      PRINT 640
C----------------
C  PUT HEADER TO FLOW TO GW IN RESERVOIR OUTPUT FILE
C----------------
      IF (NTOTRES .GT. 0) THEN
         WRITE(131,'(A)')'YEAR '
         WRITE(131,'(A)')'MONTH'
         WRITE(131,'(A)')'DAY '
         DO IA=1,NTOTRES
            WRITE(131,'(A6)') RESNAME(IA)
         ENDDO
         WRITE(131,'(A)')'DATA'
      ENDIF
C-----------------
C  PUT HEADER ON STAGE MONITOR POINTS PRINTOUT
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (NMTR .EQ. 0) GO TO 730
      DO I = 1,NMTR
        no_of_stg_outp(i) = 1
        DO k =1,n_cells(i)
          IF (noresincell(MTRP(i,k)) .GT. 0) THEN
             DO ir = 1,noresincell(MTRP(i,k))
               ia = ires_index_in_cell(MTRP(i,k),ir)
               IF (sfactor(ia) .LT. sfactmin .and.ires_small_sim(ia)
     $           .eq. 'YES') THEN
               pltnm(i,ir) = resname(ia)
               ENDIF
             ENDDO
          no_of_stg_outp(i) = noresincell(MTRP(i,k))
          ENDIF
        ENDDO
      ENDDO
      PRINT 710, ((PLTNM(I,KK),KK=1,no_of_stg_outp(i)), I = 1,NMTR)
      namelok = 'LAKOK'
      WRITE(76,'(A)')'YEAR '
      WRITE(76,'(A)')'MONTH'
      WRITE(76,'(A)')'DAY '
      WRITE(76,'(A)')'ZONE'
      DO I = 1,NMTR
        WRITE(76,'(A6)') (PLTNM(I,KK),KK=1,no_of_stg_outp(i))
      ENDDO
      WRITE(76,'(A)') namelok
      WRITE(76,'(A)')'DATA'
  730 IF (NMC.NE.0.AND.LP(8)) THEN

         WRITE(78,'(A)')'YEAR '
```

```
      WRITE(78,'(A)')'MONTH'
      WRITE(78,'(A)')'DAY '
      DO I=1,NMC
        WRITE(78,'(A5)') CNM(MTRC(I))
      ENDDO
      WRITE(78,'(A)')'DATA'
    ENDIF
    RETURN
  3 FORMAT(2(/),'CANAL NAME',2x,A5,2x,'IN CANAL_STRUC_SPECS.DAT DOES N
   $OT MATCH NAME',2x,A5,2x,'IN canal_grid_loc.dat. MAY HAVE A MISMATC
   $H. THE CANALS INPUT IN BOTH FILES MUST BE IN SAME ORDER.  CHECK FI
   $LES.')
  5 FORMAT(2(/),'NO OF DS CANALS INPUT =',I2,2x,' TO RECEIVE WATER SUP
   $PLY DISCHARGES FROM CANAL',2x,A5,2x,'DOES NOT MATCH THE NUMBER EXP
   $ECTED FOR CANAL, WHICH =',2x,I2,2x,'CHECK CANAL_STRUC_SPECS.DAT
   $FILE. FIRST,MAKE SURE THE CANAL(S) RECEIVING WATER SUPPLY DISCHARG
   $ES IS(ARE) LISTED AFTER UPSTREAM CANAL IN canal_grid_loc.dat and c
   $anal_struc_specs.dat FILES')
  6 FORMAT(2(/),'UNRECOGNIZED TRIGGER CANAL NAME',1x,A5,2x,'INPUT FOR
   $STRUCTURE',2x,A6,2x,'CHECK canal_struc_specs.dat FILE')
  7 FORMAT(2(/),'UNRECOGNIZED UPSTREAM CANAL NAME',1x,A5,2x,'INPUT FOR
   $ STRUCTURE',2x,A6,2x,'WHOSE TRIGGER IS REMOTE. CHECK canal_struc_s
   $pecs.dat FILE')
  8 FORMAT(2(/),'UNRECOGNIZED STRUC NAME',2x,A6,2x,'WHOSE TRIGGER IS R
   $EMOTE. CHECK canal_struc_specs.dat FILE')
  9 FORMAT(2(/),'UNRECOGNIZED GRID LOCATION',2x,2i5,2x,'ALONG CANAL',
   $2x,A5,2x,'TO TRIGGER FLOW THRU',2x,A6,2x,'CHECK canal_struc_specs.
   $dat FILE')
 10 FORMAT(2(/),'UNRECOGNIZED DS TARGET NAME FOR STRUC OUTF',2x,A5,2x,
   $'FOR STRUC',2x,A6,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
 17 FORMAT(2(/),'UNRECOGNIZED TRIGGER CANAL NAME',1x,A5,2x,'INPUT FOR
   $STRUCTURE',2x,A6,2x,'WHOSE ONLY TRIGGER IS REMOTE ON/OFF SWITCH. C
   $HECK canal_struc_specs.dat FILE')
 18 FORMAT(2(/),'UNRECOGNIZED STRUC NAME',2x,A6,2x,'WHOSE ONLY TRIGGER
   $ IS REMOTE ON/OFF SWITCH. CHECK canal_struc_specs.dat FILE')
 19 FORMAT(2(/),'UNRECOGNIZED GRID LOCATION',2x,2i5,2x,'ALONG CANAL',
   $2x,A5,2x,'TRIGGERING FLOW THRU',2x,A6,2x,'CHECK canal_struc_specs.
   $dat FILE')
 20 FORMAT(A5,1X,2F5.1,5F7.1,2X,A5,4I5,2x,A2)
 30 FORMAT(A6,2x,A5,2x,2(2i6,1x),f7.2,2f7.1)
 50 FORMAT(2(/),'UNRECOGNIZED RESERVOIR NAME',2x,A6,2x ,'FOR STRUC',
   $2x,A6,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
 60 FORMAT(2(/),'UNRECOGNIZED TARGET NAME FOR STRUC OUTF',2x,A5,2x,
   $'FOR STRUC',2x,A6,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
 70 FORMAT(2(/),'UNRECOGNIZED RESERV INFLOW STRUC NAME',2x,A6 ,2x,
   $'FOR OUTFLOW STRUC',2x,A6,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
 80 FORMAT(2(/),'UNRECOGNIZED DS CANAL NAME FOR LIMITING STRUC DISCHAR
   $GES',2x,A5,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
 90 FORMAT(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x ,
   $'USED AS TRIGGER FOR OUTFLOW FROM S-197')
100 FORMAT(2(/),'UNRECOGNIZED DOWNSTREAM CANAL NAME',2x,A5 ,2x,
   $'FOR WEIR',2x,A6,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE')
140 FORMAT(2(/),'UNRECOGNIZED OUTFLOW STRUC ID',2x ,A6,2x,'FOR CANAL',
   $2x,A5,2x,'CHECK CANAL_STRUC_SPECS.DAT FILE. ALSO MAKE SURE STRUC I
   $D APPEARS IN MASTER LIST OF STRUC NAMES IN MODEL DEFINITION DATA F
   $ILE.')
150 FORMAT(2(/),'UNRECOGNIZED DWNSTREAM CANAL ID',2x ,A5,2x,
```

```
     $'FOR CANAL',2x,A5,2x,'AND STRUC',2x,A6,2x,
     $'CHECK CANAL_STRUC_SPECS.DAT FILE')
  177 FORMAT(2(/),'UNRECOGNIZED UPSTREAM CANAL NAME',1x,A5,2x,'INPUT FOR
     $ STRUCTURE',2x,A6,2x,'WHOSE ONLY TRIGGER IS REMOTE ON/OFF SWITCH.
     $CHECK canal_struc_specs.dat FILE')
  230 FORMAT(2(/),'UNRECOGNIZED LEVEE NAME',2x,A5)
  240 FORMAT(3I4,2x,A5,2x,A5,1x,3i4)
  290 FORMAT(1X,'NO OF PRS:',I6)
  300 FORMAT(1X,'FROM NODE:',95(I3,',',I2))
  310 FORMAT(1X,'TO NODE  :',95(I3,',',I2))
  320 FORMAT (1H1//' NUMBER OF CANALS',T55,I4/,
     $' CHANNEL FLOW TIME STEPS PER DAY',T55,I4/,
     $' CANALS WRITTEN TO PLOT FILE',T40,10(2X,A5)/T40,10(2X,A5))
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
  344 FORMAT(A6,6x,A6,2F10.2)
  370 FORMAT (//T20,'KNOWN FLOW POINTS',/,T13,'STR',4X,'IOPT   IDV')
  400 FORMAT(2(/),'UNRECOGNIZED CANAL NAME IN known_flow_route_specs.dat
     $ FILE',2x,A5)
  410 FORMAT(11x,A6,2x,2I6,2x,A5)
  440 FORMAT(11x,A6,2x,4i6,2x,A5)
  460 FORMAT(11x,A6,2x,2i6,2x,A5,2x,2i6)
  480 FORMAT(11x,A6,2x,2i6,2(2x,A5))
  500 FORMAT(11x,A6,2x,3i6,2x,A5)
  620 FORMAT (11X,A6,3X,I3,I6,2X,4I6)
  640 FORMAT (//,T33,'KNOWN FLOW DISTRIBUTION', /,T10,
     $'IOPT = 1 : FLOW INTO PONDING    ', T60,
     $'A "*" BEFORE THE STRUCTURE NAME ', /,T10,
     $'IOPT = 2 : FLOW INTO A CANAL    ', T60,
     $'INDICATES THE FLOW VALUE WILL BE', /,T10,
     $'IOPT = 3 : FLOW FROM POND1 TO POND2      ', T60,
     $'CALCULATED BY THE ROUTING ROUTINES', /,T10,
     $'IOPT = 4 : FLOW FROM PONDING TO A CANAL  ', /,T10,
     $'IOPT = 5 : FLOW FROM CANAL TO PONDING', /,T10,
     $'IOPT = 6 : FLOW FROM ONE CANAL TO ANOTHER', //T10,
     $'DIVIDE FLOW BY IDV TO DISTRIBUTE IT')
  650 FORMAT(2(/),'WARNING:HEADWATER LOCATION',2i5,2x,'OF WEIR IN CANAL'
     $,2x,A5,2x,'IS NOT ALONG UPSTREAM CANAL. CHECK CANAL_STRUC_SPECS.DA
     $T FILE')
  660 FORMAT(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,
     $'FOR SEASONAL OPERATION OF WEIR')
  665 FORMAT(2(/),'UNRECOGNIZED ALTERNATE TRIGGER CANAL NAME',1x,A5,2x,'
     $INPUT FOR STRUCTURE',2x,A6,2x,'CHECK canal_struc_specs.dat FILE')
  667 FORMAT(2(/),'UNRECOGNIZED STRUC NAME AS TRIGGER',2x,A6,2x,'WHOSE TR
     $IGGER IS REMOTE. CHECK canal_struc_specs.dat FILE')
  670 FORMAT (///,T3,'SEASONAL REGULATION STAGES',//,T5, 'CANAL',5X,
     $'DRY',5X,'WET')
  680 FORMAT (5X,A5,12F8.2)
  710 FORMAT (1H1,//,T6,'DATE',T13,'AG-VOL',T40,19(A6,1X)/)
  750 FORMAT(2(/),'WARNING:TAILWATER LOCATION',2i5,2x,'OF WEIR IN CANAL'
     $,2x,A5,2x,'IS NOT ALONG DOWNSTREAM CANAL. CHECK CANAL_STRUC_SPECS.
     $DAT FILE')
  850 FORMAT(2(/),'WARNING:HEADWATER LOCATION',2i5,2x,'OF STRUC',2x,A6
     $,2x,'IS NOT ALONG UPSTREAM CANAL. CHECK CANAL_STRUC_SPECS.DAT FILE
     $')
  950 FORMAT(2(/),'WARNING:TAILWATER LOCATION',2i5,2x,'OF STRUC',2x,A6
     $,2x,'IS NOT ALONG DOWNSTREAM CANAL. CHECK CANAL_STRUC_SPECS.DAT FI
     $LE')
```

```
 960 FORMAT(2(/),'UNRECOGNIZED DOWNSTREAM BORROW CANAL NAME',2x,A5,2x,
    $'FOR LEVEE',2x,A5,2x,'CHECK LEVEE_SPG_INPUT.DAT FILE')
 970 FORMAT(2(/),'UNRECOGNIZED UPSTREAM BORROW CANAL NAME',2x,A5,2x,
    $'FOR LEVEE',2x,A5,2x,'CHECK LEVEE_SPG_INPUT.DAT FILE')
 980 FORMAT(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,'FOR PRE-STORM DRAW
    $DOWN. CHECK DRAWDOWN.DAT FILE')
 990 FORMAT(2(/),'UNRECOGNIZED STRUC NAME',2x,A5,2x,'FOR PRE-STORM DRAW
    $DOWN. CHECK DRAWDOWN.DAT FILE')
1000 FORMAT(2(/),'UNRECOGNIZED RESERVOIR NAME',2x,A6,2x,'FOR PRE-STORM
    $DRAWDOWN. CHECK RESERVOIR DRAWDORN DATA FILE')
1010 FORMAT(2(/),'UNRECOGNIZED RESERVOIR OUTFLOW STRUC NAME',2x,A6,2x,
    $'FOR PRE-STORM DRAWDOWN. CHECK RESERVOIR DRAWDORN DATA FILE')
1020 FORMAT(2(/),'UNRECOGNIZED NAME OF UPSTREAM CANAL',2x,A5,2x,' FOR D
    $UAL OPERATIONS. CKECK DUAL_OPS.DAT FILE')
1030 FORMAT(2(/),'UNRECOGNIZED NAME OF DESTINATION CANAL',2x,A5,2x,' FO
    $R DUAL OPERATIONS. CKECK DUAL_OPS.DAT FILE')
1040 FORMAT(2(/),'UNRECOGNIZED STRUCTURE NAME',2x,A6,2x,' FOR DUAL OPER
    $ATIONS. CKECK DUAL_OPS.DAT FILE')
1050 FORMAT(2(/),'CANAL',2x,A5,2x,'IS LISTED BEFORE CANAL',2x,A5
    $,2x,'. SINCE',2x,A5,2x,'IS RECEIVING WATER SUPPLY FROM',2x,A5,2x,
    $'PLEASE REVERSE THE ORDER IN canal_struc_specs.dat AND canal_grid_l
    $oc.dat FILES.')
     END
```

cnlneeds.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: cnlneeds.inc,v 1.5 2003/07/03 20:07:57 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/cnlneeds.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      common /cnlneeds/ tdqu_dn(max_ncnls),iseep_lim_s356_opt,s334_fc
     $,s337fc
```

cstlucie.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: cstlucie.inc,v 1.2 2003/07/03 20:07:57 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/cstlucie.inc,v $
C       $Name: rel-5-5 $
-
C
C-----------------------------------------------------------------------
      common /cstlucie/ ealc44ro,eallok1,alc44,eflc44,eallok2
     &      ,eqs80,eqs308,alsle0,c44ro0,sledmd0,s308cap0
     &      ,s80cap0,s308cap,s80cap,sledmd_met_by_lok
     &      ,eallokres2,c44dmd_met_by_lokres
```

daily_output.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: daily_output.F,v 1.15 2004/03/05 21:20:21 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/daily_output.F,v $
C   $Name: rel-5-5 $
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE daily_output (stagelo, s12_reg_total, s12_env_total,
     & rmflws333, lok_split_option, sumrain, n_cells, lo_reg_zone,
     & stagelo_res, no_of_small_res_budg_files, iunit_no_res_file,
     & deltastor, iseas, pond_end, stage_g3273,ltotal,
     & daily_seep,i_canal_rf_chg_ops)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION icnl_slp_no(max_n_cnls_w_var_slpe),
     & iunit_no_res_file(max_n_sm_res), deltastor(max_n_res),
     & n_cells(max_n_gage_loc_outp),pond_end(max_ncells),
```

```
     & no_of_stg_outp(max_n_gage_loc_outp),
     & res_resid(max_n_res), sumrain(max_ncells),
     & daily_seep(max_n_lvsp_loc),i_canal_rf_chg_ops(max_ncnls)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  output components of ASR Budget

      IF (runmode .EQ. 'SIMUL' .AND. ntotasr .GT. 0) THEN
         num_asr_output = ntotasr - stl_asr_flag - cal_asr_flag
         IF (num_asr_output .gt. 0) THEN
           WRITE (65,'(i5,2i3,5x,20(4f8.3,12X))') ifyr+iyear-1,month,
     $     iday ,(asrvol(iasrid)/1000.,total_asr_inj(iasrid)/1000. ,
     $     asrloss(iasrid)/1000.,total_asr_rec(iasrid)/1000. ,iasrid=1,
     $     num_asr_output)
         ENDIF
      ENDIF
      IF (NMTR .GT. 0 ) THEN
         DO J = 1,NMTR
            DO ii = 1,max_n_res_in_cell
              rmean_acstg(J,ii) = 0.0
            ENDDO
            DO K = 1,n_cells(j)
               IF (H(MTRP(j,k)) + POND(MTRP(J,k)) .GE.ELLS(MTRP(j,k)))
     $         THEN
                  ACSTG(J,1) = H(MTRP(j,k)) + POND(MTRP(J,k))
                  DO IR = 1,noresincell(MTRP(J,k))
                     ia = ires_index_in_cell(MTRP(J,k),ir)
                      IF (sfactor(ia) .GE. sfactmin .OR.ires_small_sim
     $                (ia) .eq. 'NO ') THEN
                         IF (H(MTRP(J,k)) + POND(MTRP(J,k)) .LE. ELLS
     $                   (MTRP(J,k))) THEN
                            AFACT = 1.0
                         ELSE
                            AFACT = SFACTOR(IA)
                         ENDIF
                         term1_for_adjust_area = (SUMET_outside_res(MTRP
     $                   (j,k))+seep_loss_res(MTRP(j,k))-SUMRAIN(MTRP(j
     $                   ,k))/12. 0- sum_chg_depth_levee_spg(MTRP(j,k)))
     $                   *((1. 0- afact)/ afact)
                         ACSTG(J,ir) = ELLS(MTRP(J,k)) + POND(MTRP(J,k))
     $                   /AFACT + term1_for_adjust_area -
     $                   (outf_fc_frm_res_to_cell(MTRP(J,k)) +
     $                   outf_seep_loc_to_res_cell(MTRP(J,k)))/afact -
     $                   sum_chg_dpth_ovl_flw(MTRP(J,k))/afact -
     $                   adjust_dur_excess(MTRP(J,k))
c
                         ACSTG(J,ir)=ACSTG(J,ir)-adjust_for_area_m1
     $                   (MTRP(J,k))
                      ELSE
                         ACSTG(J,ir) = stage_res(MTRP(j,k),ir)
                      ENDIF
c
c              IF (resname(ia).EQ.'STA1W ') print *, ACSTG(j,ir)
c    +         ,SUMET(MTRP(J,k)),SUMRAIN(MTRP(J,k))/12.0
c    +         ,adjust_for_area_m1(MTRP(J,k)),((1.0-AFACT)/AFACT)
c    +         ,outf_fc_frm_res_to_cell(MTRP(J,k))
```

```
c    +         ,outf_seep_loc_to_res_cell(MTRP(J,k))
c    +         ,sum_chg_depth_levee_spg(MTRP(J,k))
c    +         ,IBSN(MTRP(J,k)),seep_loss_res(mtrp(j,k)),POND(MTRP(J,k))
c    +         ,adjust_dur_excess(mtrp(j,k)),sum_chg_dpth_ovl_flw
c    +         (MTRP(J,k)),'    VVVV ',MTRP(j,k),month,iday,ifyr+iyear-1
c    +         ,daily_excess_adjust(MTRP(j,k))
                 END DO
               ELSE
                 ACSTG(J,1) = H(MTRP(j,k)) + POND(MTRP(J,k))
                 DO IR = 1,noresincell(MTRP(J,k))
                   ia = ires_index_in_cell(MTRP(J,k),ir)
                   IF (sfactor(ia) .LT. sfactmin.AND.ires_small_sim
     $             (ia) .eq. 'YES') THEN
                       ACSTG(J,ir) = stage_res(MTRP(j,k),ir)
                   ENDIF
                 ENDDO
               ENDIF
               IF (noresincell(MTRP(J,k)) .GT. 0) THEN
                 DO ir = 1,noresincell(MTRP(J,k))
                   rmean_acstg(J,ir) = rmean_acstg(J,ir) + ACSTG(J,ir)
     $             /n_cells(j)
                 ENDDO
                 no_of_stg_outp(j) = noresincell(MTRP(J,k))
               ELSE
                 rmean_acstg(J,1) = rmean_acstg(J,1) + ACSTG(J,1)
     $             /n_cells(j)
                 no_of_stg_outp(j) = 1
               ENDIF
            END DO
         END DO
         CALL match ('LBLTC',1,5,cnm,nch+1,5,ilbtc)
         IF (ilbtc .EQ. 0) ilbtc = nch+1
         CALL match ('LBLTN',1,5,cnm,nch+1,5,ilbtn)
         IF (ilbtn .EQ. 0) ilbtn = nch+1
         DO i = 1,NMTR
            IF (pltnm(i,1) .EQ. 'LBTCC') rmean_ACSTG(I,1)=chdep(ilbtc)
            IF (pltnm(i,1) .EQ. 'LBTNC') rmean_ACSTG(I,1)=chdep(ilbtn)
         enddo
         IF (lok_split_option .EQ. 1) THEN
           WRITE(76,10) IFYR+IYEAR-1,MONTH,IDAY,lo_reg_zone,
     $     ((rmean_ACSTG(I,IR),IR=1,no_of_stg_outp(i)),I = 1,NMTR)
     $     ,STAGELO,STAGELO_RES
         ELSE
           WRITE(76,10) IFYR+IYEAR-1,MONTH,IDAY,lo_reg_zone,
     $     ((rmean_ACSTG(I,IR),IR=1,no_of_stg_outp(i)),I = 1,NMTR)
     $     ,STAGELO
         ENDIF
      ENDIF
C
      IF (NTOTRES .GT. 0) THEN
         WRITE(131,'(i5,2i3,2x,30F7.0)') ifyr+iyear-1,month,iday,
     $   (est_flow_to_gwt(ia) ,ia=1,ntotres)
      ENDIF
c
      IF (.NOT. opt_ft_input) THEN
         WRITE(141,'(i5,2i3,2x,2f10.0)') ifyr+iyear-1,month,iday, FRF
```

```
      ENDIF
c
c  compute mean ponding for cells with separate reservoir(s)
c
      DO 250 I = MAXY,1,-1
         DO 240 NODE = ISUM(I) +  1, ISUM(I+1)
c
c  incorporate small reservoirs as part of grid system
c
            total_sfactor = 0.0
            PONDnode = 0.0
            DO ir = 1,noresincell(node)
               ia = ires_index_in_cell(node,ir)
               IF (sfactor(ia) .LT. sfactmin .AND. ires_small_sim
     $         (ia) .eq. 'YES') THEN
                  total_sfactor = sfactor(ia) + total_sfactor
                  PONDnode = pond_in_res(node,ir)*sfactor(ia)
     $            + PONDnode
               endif
            enddo
            PONDnode1 = PONDnode + POND(NODE)
     $      *(1.0 - total_sfactor - carfact(node))
            POND_END(NODE) = PONDnode1
  240    CONTINUE
  250 CONTINUE
c
c  print out water budgets for small reservoirs
c
       DO ia = 1,ntotres
          IF (sfactor(ia) .LT. sfactmin .AND. ires_small_sim(ia)
     $    .EQ. 'YES'.or. resname(ia) .eq. 'STA1E ') THEN
             DO ir = 1,noresincell(NODEWA(ia))
                IF (ia .eq. ires_index_in_cell(NODEWA(ia),ir)) THEN
                 call match (resname(ia),1,6,res_name_budg
     $           ,no_of_small_res_budg_files,6,imatch)
                 IF (imatch .ne. 0) then
                   res_resid(ia) = deltastor(ia) - res_rain(ia)
     $             - res_from_asr(ia) - res_struc_inflw(ia)
     $             + res_et(ia) + res_seep_out(ia) + res_to_asr(ia)
     $             + res_struc_outflw(ia) + res_lvseep_out(ia)
                   write(iunit_no_res_file(imatch),100) ifyr+iyear-1
     $             ,month,iday,stage_res(NODEWA(ia),ir),h(NODEWA(ia))
     $             +pond(NODEWA(ia))+solmx(nodewa(ia)),res_rain(ia)
     $             ,res_from_asr(ia),res_struc_inflw(ia),res_et(ia)
     $             ,res_seep_out(ia),res_struc_outflw(ia)
     $             ,res_lvseep_out(ia),res_to_asr(ia),res_resid(ia)
                 ENDIF
               ENDIF
            ENDDO
         ENDIF
      ENDDO
 100  FORMAT(I5,2I3,2X,2F7.2,10F7.0)
c
c
c  print out NSM flow target,S12,S333,S355,if applicable
c
      IF (type_flow_across_ttrail .NE. 'MINDEL ') THEN
```

```
         IF (type_flow_across_ttrail .NE. 'STAGETG') THEN
            WRITE(151,'(i5,2i3,5x,10f7.0,3x,A1,2x,I1,F7.2)') ifyr+iyear
     $      - 1,month,iday,s12_flow_targ,RMFLWS333,s12_reg_total ,
     $      s12_env_total,s333_flow_targ,s333_reg,s333_env ,
     $      s355_flow_targ,s355_reg,s355_env,wca3zone,ig3273_const_indx,
     $      stage_g3273

         ENDIF
      ENDIF
c
c  print out daily slopes for appropriate canals
c
      do kk = 1,ncnl_dly_slope
        call match (canal_name_dvslope(kk),1,5,cnm,nch+1,5,imatch)
        icnl_slp_no(kk) = imatch
      enddo
      write(190,'(i5,2i3,2x,50F6.2)') ifyr+iyear-1,month,iday
     $  ,(HDC(icnl_slp_no(ll),iseas),ll=1,ncnl_dly_slope)
      write(290,'(i5,2i3,2x,50i6)') ifyr+iyear-1,month,iday
     $  ,(i_canal_rf_chg_ops(icnl_slp_no(ll)),ll=1,ncnl_dly_slope)
      DO ia = 1,ntotres
         res_struc_inflw(ia) = 0.0
      ENDDO
C
C  print out levee seepage
c
      IF (lp(9)) THEN
         WRITE(102,180)IFYR+IYEAR - 1,MONTH,iday,(daily_seep(L),L = 1,
     $   LTOTAL)
      ENDIF
C
180   FORMAT(3x,i5,2(1x,i2),90f9.2)
      RETURN
   10 FORMAT(I5,3I3,80(F6.2,1X))
      END
```

daily_ovlnf_out.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: daily_ovlnf_out.F,v 1.8 2003/07/03 20:07:20 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/daily_ovlnf_out.F,v $
C   $Name: rel-5-5 $
C
C
C   PRINT OUT THE DESIRED ARRAYS WHEN CALLED BY MAIN PROGRAM
C   PRINTING OCCURS WHEN LP INDEX IS TRUE
C   VARIABLES:
C    AGP       TOTAL PUMPAGE IN AG AREA
C    B         CHAR. STRING TO BLANK FILL WHEN MINX # 1
C    COAST     SUM OF COASTAL CANAL DISCHARGE
C    IDD       DAY OF MONTH
C    IDYMAX    NUMBER OF DAYS IN MONTH
C    LBOT      LAST LINE OF ARRAY TO BE PRINTED
C    LP        ARRAY OF LOGICAL PRINT INDICATORS
C    MTRC      NUMBER OF CANAL MONITORING POINTS
C    NMTR        "       STAGE     "         "
C    MTRC      SUBSCRIPT VALUES OF CANAL MONITOR POINTS
C    MTRP          "        "      STAGE    "      "
C    NCH       NUMBER OF CANALS IN MODEL
C    OLD       ARRAY USED FOR MISCELLANEOUS LISTINGS
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE daily_ovlnf_out (idd, runtitle)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
```

```
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'daydump.inc'

      CHARACTER datetag*256, runtitle*256

      INTEGER ierr

      REAL values(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX, MAXX,
     +ISUM)

C  MAKE A DATE TAG FOR BINARY FILES

      IF (iddump) THEN

C  output daily values of surface water flow
C  IT = 1, FLOW IN THE X DIRECTION

         CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1,datetag)
         DO NODE = 1, ISUM(MAXY + 1)
            VALUES(NODE) = OVLFLO_daily(NODE, 1)/43560.
         END DO
         CALL GWRITE(43,datetag, VALUES, IERR)
C
C           IT = 2, FLOW IN THE Y DIRECTION
C           NOTE Y FLOW IS COMPUTED IS POSITIVE TO SOUTH, SHOULD
C           BE POSITIVE TO NORTH
C
         DO NODE = 1, ISUM(MAXY + 1)
            VALUES(NODE) = - OVLFLO_daily(NODE, 2)/43560.
         END DO
         CALL GWRITE(43,datetag, VALUES, IERR)
      ENDIF
      RETURN
      END
```

daily_variables_init.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: daily_variables_init.F,v 1.16 2005/06/10 13:29:02 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/daily_variables_init.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine initializes appropriate variables on a daily basis.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE daily_variables_init (total_flow_g404, rfls155a,
     & s31dsg_to_bay, total_reuse_res, add_flow_to_sta34,
     & ws_from_acmres_lwdd, ws_to_sta34_frm_lok, water_supply_to_holyl,
     & tot_flw_to_c9r_frm_nlkblt, tot_lok_to_eaares, rmflws12,rmflws333,
     & s333_reg_lec, asr_rec_ws, asr_rec_ws_cnty, desgn_cap_canl_ws,
     & total_reuse_ws, temp_volwtr, volwtr, mod_delta_stor,
     & depth_levee_spg_max_from_gw, bmp_makeup_water,
     & rmin_lok_stg_del_est, day_pet0, nyday, tempf_volwtr,
     & day_pet0_lok, nsvarea, ntotal_no_import_areas, jday,
     & lok_split_option, fract_mds_res, fract_mds_litzone, bmp_opt,
     & isem_flg, sem_dmnd_orig, total_et_red, ws_to_l8_via_s352,
     & vol_rec_from_cpbasr, cfsoth_res, cfsoth_litz, numd, iyr,
     & s333_reg_lec_abve_sch, s333_reg_lec_bel_sch, vol_rec_from_acmasr,
     & runoff_to_sta,rlokreg_to_l8)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'simcstgs.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
```

```
      INCLUDE 'et.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'daydump.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION asr_rec_ws(max_n_serv_area),
     & asr_rec_ws_cnty(max_n_serv_area), bmp_makeup_water(366),
     & day_pet0(max_ncells,31), day_pet0_lok(31),
     & depth_levee_spg_max_from_gw(max_ncells),
     & desgn_cap_canl_ws(max_n_eaa_condt), numd(12), rmds_litzone(366),
     & rmds_res(366), temp_volwtr(max_n_res), tempf_volwtr(max_n_res),
     & total_reuse_ws(max_n_serv_area), volwtr(max_n_res)

      CHARACTER bmp_opt*5

      INTEGER phase, trig, zone

      REAL mod_delta_stor(366)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C Initialize variables used on a Daily Basis
C Many of the variables are for accounting purposes only

      iddump = .FALSE.
      IF (lp(19) .or. lp(25)) THEN
         ijul = iymdjl(iyr,month,iday)
         IF ( (ijul1.le.ijul) .AND. (ijul.le.ijul2) ) iddump = .TRUE.
      ENDIF

      IDSS=IDSS+1
      JSTR=0
      total_et_red = 0.0
      s5a2_so = 0.0
      s5a2_no = 0.0
      s5a3_no = 0.0
      s5a3_so = 0.0
      s355_reg = 0.0
      s355_env = 0.0
      s31dsg_to_bay = 0.0
      g136_to_sta34 = 0.0
      rlokreg_to_l8 = 0.0
      rmiami_runoff_to_sta = 0.0
      rnnrc_runoff_to_sta = 0.0
```

```
eaa_runff_s5a_to_sta = 0.0
runoff_to_sta = 0.0
ws_to_l8_via_s352 = 0.0
rfls155a = 0.0
total_reuse_res = 0.0
cout_to_c51_frm_l8bprs = 0.0
outflow_to_l8s = 0.0
rl8runoff_to_bptres = 0.0
flow_to_l8sta = 0.0
add_flow_to_sta34 = 0.0
outflow_from_roten_to_semcyp = 0.0
ws_from_acmres_lwdd = 0.0
total_flow_g404 = 0.0
inflow_to_sta6_frm_unit1 = 0.0
emerg_bypass_sta34   = 0.0
flow_to_s7_sta2byp   = 0.0
holy_out_to_s140    = 0.0
flow_to_sta2_frm_eaa = 0.0
ws_to_sta34_frm_lok   = 0.0
water_supply_to_holyl = 0.0
ws_from_acmres_cpbres = 0.0
outflow_from_us27s = 0.0
outflow_from_us27n = 0.0
vol_rec_from_c51asr  = 0.0
vol_rec_from_hlsbasr = 0.0
vol_rec_from_nnrcasr = 0.0
vol_rec_from_cpbasr  = 0.0
vol_rec_from_acmasr  = 0.0
vol_rec_from_l31asr  = 0.0
ws_frm_res_to_wca1   = 0.0
ws_frm_res_to_wca2a  = 0.0
ws_frm_res_to_wca3a  = 0.0
tot_flw_to_c9r_frm_nlkblt = 0.0
s333_reg_lec_abve_sch = 0.0
s333_reg_lec_bel_sch = 0.0
tot_lok_to_eaares = 0.0
runoff_to_lec_s6 = 0.0
runoff_to_lec_s7 = 0.0
runoff_to_lec_s8 = 0.0
s12_flow_targ  = 0.0
s333_flow_targ = 0.0
s355_flow_targ = 0.0
RMFLWS12 = 0.0
S331WS   = 0.0
S331FC   = 0.0
RMFLWS333 = 0.0
s333_reg_lec = 0.0
REGS31 = 0.0
WSS31  = 0.0
nlsc = 0
DO n = 1,nch
   upstream_inflow(n) = 0.0
   IF (ivary_slope_daily_opt(n).eq.1) THEN
     nlsc = nlsc + 1
     DO kk = 1,2
       DO ll = 1,3
         dischg_for_slope_kn(nlsc,kk,ll) = 0.0
```

```
            ENDDO
          ENDDO
        ENDIF
      END DO
      DO isrvcarea = 1,nsvarea
        asr_rec_ws(isrvcarea) = 0.0
        asr_rec_ws_cnty(isrvcarea) = 0.0
      END DO
      DO i = 1,max_n_wcas
        asr_rec_vol_wca(i) = 0.0
      END DO
      DO i = 1,neaacnl
        cnlcap_ws(i) = desgn_cap_canl_ws(i)
      END DO
      DO i = 1,nsvarea
        total_reuse_ws(i) = 0.0
      END DO
      IF ((month.eq.11 .or. month .eq. 6) .and. iday .eq. 1) then
        DO nn = 1,nch
          IF (ivary_slope_daily_opt(nn) .eq. 1) then
              if (month.eq.11) then
                hdc(nn,iseas) = hdc(nn,2)
              elseif (month.eq.6) then
                hdc(nn,iseas) = hdc(nn,1)
              endif
          ENDIF
        ENDDO
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (NTOTRES .GT. 0) THEN
        DO IA = 1,NTOTRES
          QSTAIN(IA) = 0.0
          daily_inflow_to_res_frm_cnl(ia) = 0.0
          est_flow_to_gwt(ia) = 0.0
          temp_volwtr(ia) = volwtr(ia)
          tempf_volwtr(ia) = volwtr(ia)
          res_rain(ia) = 0.0
          res_et(ia) = 0.0
          res_struc_outflw(ia) = 0.0
          res_seep_out(ia) = 0.0
          res_lvseep_out(ia) = 0.0
          res_to_asr(ia) = 0.0
          res_from_asr(ia) = 0.0
        END DO
      ENDIF
      DO isa = 1,ntotal_no_import_areas+1
        LFLWTRG(ISA) = .TRUE.
      END DO
c
      DO NODE = 1,ISUM(MAXY+1)
        DO IT = 1,2
          OVLFLO_daily(NODE,IT) = 0.0
        END DO
        depth_levee_spg_max_from_gw(NODE) = 0.0
        daily_gw_loss_cell(NODE) = 0.0
        et_sat(NODE) = 0.0
        et_ponding(NODE) = 0.0
```

```
      END DO
c
      DO ia = 1,nmarea
         DO j = 1,ninstr(ia)
            qinflw(ia,j) = 0.0
         END DO
      END DO
c
      DO j = 1,ninstr(indxsta(10))
         QINFLW(indxsta(10),j) = 0.0
      END DO
c
      DO i = 1,100
         ncount(i) = 0
      END DO
c
      if (runmode .ne. 'CALIB') kflo(iflbel5) = 0.0
      if (runmode .ne. 'CALIB') kflo(ifll8)   = 0.0
C
C-----------------------------------------
C     COMPUTE JULIAN DAY
C-----------------------------------------
      JDAY = JULIAN(MONTH,IDAY,NUMD)
      IF (isem_flg .EQ. 1) THEN
         sem_dmnd_orig = sem_cyp_dmnd(jday)
      ENDIF
C
      IF ((MONTH .NE. 2 .OR. IDAY .NE. 29) .AND. (MONTH .NE. 12 .OR.
     $IDAY .NE. 31)) THEN
         NYDAY = NYDAY + 1
      ENDIF
c
c  initialize available capacity for injection into and
c  recovery from ASR wells
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      DO iasr = 1,ntotasr
         rem_incap(iasr) = asr_incap(iasr)
         IF (asrvol(iasr) .GE. rmin_bubble_sze_add_rec(iasr)) THEN
            rec_asr_outcap(iasr) = asr_outcap(iasr)
         ELSE
            rec_asr_outcap(iasr) = asr_incap(iasr)
         ENDIF
         rem_asr_outcap(iasr) = rec_asr_outcap(iasr)
         total_asr_inj(iasr) = 0.0
         total_asr_rec(iasr) = 0.0
      END DO
c
C
c modied delta storage
c
      cfsoth = mod_delta_stor(jday)
      IF (lok_split_option .EQ. 1) THEN
         rmds_res(jday) = fract_mds_res * mod_delta_stor(jday)
         rmds_litzone(jday) = fract_mds_litzone * mod_delta_stor(jday)

         cfsoth_res = rmds_res(jday)
         cfsoth_litz = rmds_litzone(jday)
```

```
      ENDIF
c
c Makeup water
c
      IF (make_up_water_opt .EQ. 'MAKEUP ') THEN
         bmpvol = bmp_makeup_water(jday)
      ELSE
         bmpvol = 0.0
      ENDIF
      darefet = day_pet0_lok(iday)
c
c Runoff reduction in EAA basins due to BMPs
c
      IF (bmp_opt .EQ. 'FALSE') THEN
         DO i = 1,neaabsn-1
            diff_runoff_bmp(i,jday) = 0.0
         END DO
      ENDIF
c
c  minimum LOK stage for meeting Estuarine demands
c
      CALL bpts(iday,month,10.0,1,n_fc_zones+7, 1,rmin_lok_stg_del_est)
c
c   stage in grid cells at beginning of time step
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      DO node = 1,isum(maxy)
         head_cell_beg(node) = h(node) + pond(node)
      END DO
c-----------------------
c   map daily Penman-Monteith PET (referenced to grass) from
c   the day_pet0 array back to the original potet array.
c   Note: monthly totals are computed earlier. See code after loop 30.
c-----------------------
c   change potet array to be node based
      DO i = 1,maxy
         DO node = isum(i)+1, isum(i+1)
            potet(node) = day_pet0(node,iday)
         END DO
      END DO
c
c Initialize sum_head() array if we're on the first day of the month
      if(use_trigger .GE. 1 .AND. iday.EQ.1) THEN
         DO zone = 1,n_zone
            DO trig = 1,n_trigger(zone)
               DO phase = 1,max_n_phase
                  sum_head(zone,trig,phase) = 0.0
               END DO
            END DO
         END DO
      ENDIF
      RETURN
      END
```

```
daydump.inc
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: daydump.inc,v 1.4 2003/07/03 20:07:58 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/daydump.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
C
      LOGICAL iddump
      COMMON /daydump/ iddump,ijul1,ijul2
```

dlycslope.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: dlycslope.inc,v 1.4 2003/07/03 20:07:58 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/dlycslope.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      common /dlycslope/ ivary_slope_daily_opt(max_ncnls),
     &mid_point_canl(max_ncnls),
     &dischg_for_slope_kn(max_n_cnls_w_var_slpe,2,3),ncnl_dly_slope,
     &canal_name_dvslope(max_n_cnls_w_var_slpe),
     &dischg_for_slope(max_n_cnls_w_var_slpe,2,3),
     &igrid_loc_upslp(max_n_cnls_w_var_slpe),
     &igrid_loc_dnslp(max_n_cnls_w_var_slpe),
     &icanal_swint_dep(max_ncnls),deten_cnl(max_ncnls,max_ncnlloc),
     &ofm_canl(max_ncnls,max_ncnlloc,4)
C------------------------
      CHARACTER canal_name_dvslope*5
```

dstring.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: dstring.F,v 1.5 2003/07/03 20:07:21 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/dstring.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBRROUTINE DSTRING:
C     MAKES A GOOD LOOKING DATE STRING FROM THE MONTH, DAY, YEAR VALUES
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE dstring (im, id, iy, buffer)

      CHARACTER*(*) buffer
      CHARACTER*20 jmonth(12)

      INTEGER id, im, iy

      DATA jmonth /'January', 'February', 'March', 'April', 'May',
     & 'June', 'July', 'August', 'September', 'October', 'November',
     & 'December' /

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      WRITE(BUFFER, 10) JMONTH(IM)(1:LENGTH(JMONTH(IM))), ID, IY
      RETURN
   10 FORMAT(A, 1X, I2, ', ', I4)
      END
```

dualops.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: dualops.inc,v 1.4 2003/07/03 20:07:58 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/dualops.inc,v $
C       $Name: rel-5-5 $
C
C-----------------------------------------------------
      common /dualops/
     &struc_name_dual_ops(max_n_cnl_dual_ops,max_n_dual_cond),
     &icanal_no_dualops(max_n_cnl_dual_ops,max_n_dual_cond),
     &istruc_no_dualops(max_n_cnl_dual_ops,max_n_dual_cond),
     &rmxstg_dual_ops(max_n_cnl_dual_ops,max_n_dual_cond),
     &rmnstg_dual_ops(max_n_cnl_dual_ops,max_n_dual_cond),
     &ndual_canal_ops(max_n_dual_cond),
     &ndual_col_dest(max_n_cnl_dual_ops,max_n_dual_cond),
     &ndual_row_dest(max_n_cnl_dual_ops,max_n_dual_cond),
     &ndual_canl_dest_indx(max_n_cnl_dual_ops,max_n_dual_cond),
     &igrid_loc_dest_dualops(max_n_cnl_dual_ops,max_n_dual_cond),
     &IDNCNL2(max_ncnls,max_nostr_cnl),
     &IRCEIVE2(max_ncnls,max_nostr_cnl),
     &NODRCV2(max_ncnls,max_nostr_cnl,max_n_cells_rec),
     &rmnstg_dual_ops0(max_ncnls,max_nostr_cnl)
C-------------------------------------------------------
      CHARACTER struc_name_dual_ops*6
```

eaa_caps.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: eaa_caps.inc,v 1.4 2003/07/03 20:07:58 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaa_caps.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
c COMMON block for eaa neutral case canal conveyance capacities
      common /eaa_caps/ eaa_convey(10,7),s8_tailw,s7_tailw,s5a_tailw,
     &s6_tailw,s150_tailw
```

eaa_flow_distrib_capac_setup.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: eaa_flow_distrib_capac_setup.F,v 1.8 2005/06/10 13:29:02 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaa_flow_distrib_capac_setup.F,v
$
C   $Name: rel-5-5 $
C
C-----------------------------------------------------------------------------
C   THIS SUBROUTINE SETS UP THE PARAMETERS NECESSARY FOR COMPUTATION OF
C   DISTRIBUTION OF FLOW TO OR FROM EVERGLADES AGRICULTURAL AREA.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE eaa_flow_distrib_capac_setup (jday, runoff_avg,
     & ifacteaa, rmonthly_loss_bmp, max_days_mean_bkpump, sum_alloc_ssm,
     & sum_losa_dmnd, ssmminfrac, igravity_s8_nosta, frac_eaa_dmnd_met,
     & sum_eaa_dmnd_ssm, indx_lok_del, alocnnr_potdel, pdsi,
     & frac_dmnd_met_wn, frac_dmnd_met_wt, nbrkpt_ssmwt, nbrkpt_ssmwn,
     & iregjulwt, iregjulwn, regstgwt, regstgwn, iday_of_week, totrun,
     & pfrunf)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'eaa_caps.inc'
      INCLUDE 'eaa_pump_caps.inc'
      INCLUDE 'wcasim.inc'
```

```
      INCLUDE 'opersched.inc'
      INCLUDE 'add_misc.inc'

      DIMENSION alocnnr_potdel(2), iregjulwn(25), iregjulwt(25),
     & pdsi(12), pfrunf(max_n_eaa_condt), regstgwn(25), regstgwt(25),
     & runoff(7,2), runoff_avg(2), sum_eaa_dmnd_ssm(max_n_eaa_condt)

      REAL iregjulwn, iregjulwt

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C    define counter for running average of daily runoff
C    for flood control backpumping calculations

      nmdays=nmdays+1
      IF (nmdays.gt.max_days_mean_bkpump) nmdays=max_days_mean_bkpump

      IF (day1.or.nmdays.eq.max_days_mean_bkpump) THEN
        DO i=1,2
          runoff_avg(i)=0.0
        END DO
      END IF

C  determine fraction of EAA demand to be met during the days of
C  delivery from LOK

      iday_of_week=idofw(month,iday,ifyr+iyear-1)
      sch=sched(iday,month)
      DO idw=1,n_days_week_del_eaa(indx_lok_del)
        IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(idw,indx_lok_del))
     &    THEN
          IF (nint(sum_losa_dmnd).gt.0) THEN
            frac_eaa_dmnd_met=sum_alloc_ssm/sum_losa_dmnd
            frac_eaa_dmnd_met=amin1(frac_eaa_dmnd_met,1.0)
          ELSE
            frac_eaa_dmnd_met=1.0
          END IF
          frac_eaa_dmnd_met=amax1(frac_eaa_dmnd_met,ssmminfrac)
          sum_alloc_ssm=0.0
          sum_losa_dmnd=0.0
          ssmwatch_l=schedwt(iday,month,nbrkpt_ssmwt,iregjulwt,regstgwt)
          ssmwarn_l=schedwn(iday,month,nbrkpt_ssmwn,iregjulwn,regstgwn)
          rlok_stg_dev_ssm=sch-stagelo
          IF (stagelo.ge.sch) THEN
            IF (rlok_stg_dev_ssm.gt.pdsi(month)) THEN
              IF (stagelo.lt.ssmwarn_l) THEN
                frac_eaa_dmnd_met=frac_dmnd_met_wn
              ELSE IF (stagelo.le.ssmwatch_l) THEN
                frac_eaa_dmnd_met=frac_dmnd_met_wt
              ELSE
                frac_eaa_dmnd_met=1.0
              END IF
            ELSE
              frac_eaa_dmnd_met=1.0
            END IF
          END IF
        END IF
```

```
      END DO
C
C   CALCULATE TOTAL RUNOFF FROM EAA.
C   APPROXIMATE THE PORTION OF TOTAL RUNOFF
C   THROUGH S-8 AND S3,S7+S6 AND S2,HGS5
C   AND S5A FROM HISTORICAL DATA (1979-1990)
C
      IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
        totrun=0.0
        totpct=0.0
        DO 10 i=1,neaabsn-1
          totrun=totrun+abs(amin1(agp(i),0.0))/86400.
          IF (agp(i).lt.0.0) THEN
            totpct=totpct+pctrunf(i)
          END IF
   10   CONTINUE
C
        DO 20 i=1,neaabsn-1
          IF (agp(i).lt.0.0) THEN
            pfrunf(i)=pctrunf(i)/totpct
          ELSE
            pfrunf(i)=0.0
          END IF
   20   CONTINUE
      END IF
C
      IF (runmode.eq.'SIMUL') THEN
C
C   determine  moving average of daily runoff for calculation
C   of amount of flood control backpumping to lake okeechobee
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
        DO iagbsn=1,2
          q=agp(iagbsn)/86400.
          IF (agp(iagbsn).ge.0.0) THEN
            IF (ndays_sim.ge.max_days_mean_bkpump) THEN
              DO id=1,nmdays-1
                IF (ndays_sim.gt.max_days_mean_bkpump) THEN
                  runoff(id,iagbsn)=runoff(id+1,iagbsn)
                END IF
C
                runoff_avg(iagbsn)=runoff_avg(iagbsn)+runoff(id,iagbsn)/
     &           nmdays
              END DO
              runoff(nmdays,iagbsn)=0.0
            ELSE
              runoff(nmdays,iagbsn)=0.0
              runoff_avg(iagbsn)=runoff(nmdays,iagbsn)
            END IF
            flow_reduction_bmp(iagbsn)=diff_runoff_bmp(iagbsn,jday)/
     &       1.9835+excess_reduction(iagbsn)
            excess_reduction(iagbsn)=flow_reduction_bmp(iagbsn)
          ELSE
            IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
              q=pfrunf(iagbsn)*totrun*bmpred
            ELSE
              q=abs(q)*bmpred
```

```
            END IF
            flow_reduction_bmp(iagbsn)=diff_runoff_bmp(iagbsn,jday)/
     &       1.9835+excess_reduction(iagbsn)
            excess_reduction(iagbsn)=amax1(flow_reduction_bmp(iagbsn)-q,
     &       0.0)
            rmonthly_loss_bmp=rmonthly_loss_bmp+
     &       amin1(flow_reduction_bmp(iagbsn),q)*1.9835
            q=q-amin1(flow_reduction_bmp(iagbsn),q)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C
C   determine  moving average of daily runoff for calculation
C   of amount of flood control backpumping to lake okeechobee
C
            IF (ndays_sim.ge.max_days_mean_bkpump) THEN
              DO id=1,nmdays-1
                IF (ndays_sim.gt.max_days_mean_bkpump) THEN
                  runoff(id,iagbsn)=runoff(id+1,iagbsn)
                END IF
C
                runoff_avg(iagbsn)=runoff_avg(iagbsn)+runoff(id,iagbsn)/
     &           nmdays
              END DO
              runoff(nmdays,iagbsn)=q
              runoff_avg(iagbsn)=runoff_avg(iagbsn)+q/nmdays

            ELSE
              runoff(nmdays,iagbsn)=q
              runoff_avg(iagbsn)=runoff(nmdays,iagbsn)
            END IF
          END IF
        END DO
C
C  COMPUTE DISCHARGE CAPACITIES,FUNCTION OF MAXIMUM
C  ALLOWABLE GATE OPENING,LAKE OKEECHOBEE STAGE,
C  AND TAILWATER
C
C         S-351
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
        IF (istaopt(is7bsn-6).eq.0) THEN
          ireach=2
          IF (.not.lforpump(is7bsn-6)) THEN
            istype=1
            istr_eaa=6
            rmax_cuhw=rmaxstg(is7bsn-6)
            chw=stagelo
            suhw1=stagelo
          ELSE
            istype=2
            istr_eaa=10
            rmax_cuhw=amin1(rmaxstg(is7bsn-6),stagelo+rmax_lift(is7bsn-
     &       6))
            chw=rmax_cuhw
            suhw1=rmax_cuhw
          END IF
          CALL qmax_two_reaches (suhw1, s7_tailw, istr_eaa, 2, ireach,
```

```
  &     2, grf2amx, s351cap, nnrc_cap_upper_grav, s7_cap, chw, ctw,
  &     ifacteaa)
        fllks6=grf2amx-nnrc_cap_upper_grav
        grf2amx=nnrc_cap_upper_grav
        s351_tw=chw
        s7_spill_hw=ctw
        total_neutral_capac=fllks6+grf2amx
        s351cap_neut=s351cap
        s351_tw_neut=s351_tw
        qconvey_hlsb=fllks6
        qconvey_nnrc=grf2amx
        IF (eaa_conv_opt_s7.eq.'PUMP') THEN
          chw=rmax_cuhw
          ctw=pmp_int(is7bsn-6)
          ctw2=pmp_int(is6bsn-6)
          delta0=(chw-ctw)/25.0
          delta1=delta0
          iter1=0
          tolerq=1.
          itermax=100
          s351cap_pump=-1.0
          CALL eaaspill (s351cap_pump, stagelo, chw, istr_eaa,
  &        ifacteaa)
          qcanal1=-1.0
          IF (chw.gt.ctw) THEN
            CALL eaa_prof_lu_table (qcanal1, chw, ctw, 2, ifacteaa)
          ELSE
            qcanal1=0.0
          END IF
          qcanal2=-1.0
          IF (chw.gt.ctw2) THEN
            CALL eaa_prof_lu_table (qcanal2, chw, ctw2, 3, ifacteaa)
          ELSE
            qcanal2=0.0
          END IF
          qcanal_total=qcanal1+qcanal2
          IF (s351cap_pump-qcanal_total) 30,60,40
30        sign=-1.0
          GO TO 50
40        sign=1.0
50        chw=chw+sign*delta1
          s351cap_pump=-1.0
          CALL eaaspill (s351cap_pump, stagelo, chw, istr_eaa,
  &        ifacteaa)
          qcanal1=-1.0
          IF (chw.gt.ctw) THEN
            CALL eaa_prof_lu_table (qcanal1, chw, ctw, 2, ifacteaa)
          ELSE
            qcanal1=0.0
          END IF
          qcanal2=-1.0
          IF (chw.gt.ctw2) THEN
            CALL eaa_prof_lu_table (qcanal2, chw, ctw2, 3, ifacteaa)
          ELSE
            qcanal2=0.0
          END IF
          qcanal_total=qcanal1+qcanal2
```

```
            iter1=iter1+1
            IF (iter1.eq.itermax) GO TO 60
C
            IF (sign*(s351cap_pump-qcanal_total).gt.tolerq) THEN
              GO TO 50
            END IF
C
            IF (sign*(qcanal_total-s351cap_pump).gt.tolerq) THEN
              chw=chw-sign*delta1
              delta1=delta1/2.
              GO TO 50
            END IF
   60       nnrc_cap_upper_pump=qcanal1
            fllks6_pump=qcanal2
            s351_tw_pump=chw
            s351cap_neut=s351cap_pump
            s351_tw_neut=s351_tw_pump
            s351cap=s351cap_neut
            s351_tw=s351_tw_neut
            grf2amx_pump=nnrc_cap_upper_pump
            grf2amx=grf2amx_pump
            s7_spill_hw=ctw
            total_neutral_capac=grf2amx_pump+fllks6_pump
            qconvey_hlsb=fllks6_pump
            qconvey_nnrc=grf2amx_pump
          END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          s6_hw=pmp_int(is6bsn-6)
          dschg_to_meet_s351_dmnd=0.0
          DO iw=1,n_days_week_del_eaa(indx_lok_del)
            IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &       indx_lok_del)) THEN
              dschg_to_meet_s351_dmnd=sum_eaa_dmnd_ssm(is7bsn-6)*
     &         frac_eaa_dmnd_met
            END IF
          END DO
          capac_limit=amax1(total_neutral_capac,cnlcap_ws(innr)+
     &     cnlcap_ws(ihill))
          dschg_to_eaa=amin1(dschg_to_meet_s351_dmnd,capac_limit)
          IF (dschg_to_eaa.gt.s351cap_neut.or.dschg_to_meet_s351_dmnd.gt
     &     .total_neutral_capac) THEN
            IF (.not.lforpump(is7bsn-6)) THEN
              chw=-1.0
              CALL eaaspill (dschg_to_eaa, stagelo, chw, 6, ifacteaa)
              chw=amax1(chw,rmax_tw_eaad(is7bsn-6))
              IF (chw.eq.rmax_tw_eaad(is7bsn-6)) THEN
                dschg_to_eaa=-1.0
                CALL eaaspill (dschg_to_eaa, stagelo, chw, 6, ifacteaa)
              END IF
              s351_tw_neut=chw
              ctw=-1.0
              ratio_ws=dschg_to_eaa/dschg_to_meet_s351_dmnd
              supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del)
     &         ,0.0)*frac_eaa_dmnd_met
              rem_pot_supply_to_nnrhl=dschg_to_eaa-supply_to_nnrc

              rem_pot_supply_excess_hillcap=
```

```
     &         amax1(rem_pot_supply_to_nnrhl-cnlcap_ws(ihill),0.0)
              dmnd_nnr=supply_to_nnrc+rem_pot_supply_excess_hillcap
              CALL eaa_prof_lu_table (dmnd_nnr, s351_tw_neut, ctw, 2,
     &         ifacteaa)
              nnrc_cap_upper_grav=dmnd_nnr
              IF (eaa_conv_opt_s7.eq.'PUMP') nnrc_cap_upper_pump=
     &         dmnd_nnr
C
C  hillsboro canal
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              IF (nint(dschg_to_meet_s351_dmnd).gt.0) THEN
                dmnd_hill=dschg_to_eaa-dmnd_nnr-amin1(alocnnr,0.0)/
     &           86400.
              ELSE
                dmnd_hill=0.0
              END IF
              dmnd_hill=amax1(dmnd_hill,0.0)
              dischg_to_hill=dmnd_hill
              cutw=-1.0
              CALL eaa_prof_lu_table (dischg_to_hill, s351_tw_neut,
     &         cutw, 3, ifacteaa)
              s6_hw=cutw
              fllks6=dischg_to_hill
              s351cap=dischg_to_hill+dmnd_nnr
              s351_tw=chw
              s7_spill_hw=ctw
            ELSE
              dschg_to_eaa=amin1(dschg_to_meet_s351_dmnd,
     &         total_neutral_capac)
              IF (nint(dschg_to_meet_s351_dmnd).gt.0) THEN
                ratio_ws=dschg_to_eaa/dschg_to_meet_s351_dmnd
                supply_to_nnrc=ratio_ws*
     &           amax1(alocnnr_potdel(indx_lok_del),0.0)*
     &           frac_eaa_dmnd_met
                rem_pot_supply_to_nnrhl=amax1(dschg_to_eaa-
     &           supply_to_nnrc,0.0)
                rmax_cap_ws=qconvey_hlsb
                rem_pot_supply_excess_hillcap=
     &           amax1(rem_pot_supply_to_nnrhl-rmax_cap_ws,0.0)
                dmnd_nnr=supply_to_nnrc+rem_pot_supply_excess_hillcap
              ELSE
                dmnd_nnr=0.0
              END IF
              total_nnrc_supply=dmnd_nnr
              nnrc_cap_upper_grav=amax1(qconvey_nnrc,total_nnrc_supply)
              IF (eaa_conv_opt_s7.eq.'PUMP') nnrc_cap_upper_pump=
     &         amax1(qconvey_nnrc,total_nnrc_supply)
              s7_spill_hw=ctw
            END IF
          ELSE
            IF (lforpump(is7bsn-6)) THEN
              dschg_to_eaa=amin1(dschg_to_meet_s351_dmnd,
     &         total_neutral_capac)
            END IF
            IF (nint(dschg_to_meet_s351_dmnd).gt.0) THEN
              ratio_ws=dschg_to_eaa/dschg_to_meet_s351_dmnd
```

```
              supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del)
     &         ,0.0)*frac_eaa_dmnd_met
              rem_pot_supply_to_nnrhl=amax1(dschg_to_eaa-supply_to_nnrc,
     &         0.0)
              rmax_cap_ws=amax1(cnlcap_ws(ihill),qconvey_hlsb)
              rem_pot_supply_excess_hillcap=
     &         amax1(rem_pot_supply_to_nnrhl-rmax_cap_ws,0.0)
              dmnd_nnr=supply_to_nnrc+rem_pot_supply_excess_hillcap
            ELSE
              dmnd_nnr=0.0
            END IF
            total_nnrc_supply=dmnd_nnr
            nnrc_cap_upper_grav=amax1(qconvey_nnrc,total_nnrc_supply)
            IF (eaa_conv_opt_s7.eq.'PUMP') nnrc_cap_upper_pump=
     &       amax1(qconvey_nnrc,total_nnrc_supply)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          END IF
        ELSE
          IF (eaa_conv_opt_s7.eq.'PUMP') THEN
            s351_tw=eaa_convey(5,3)
          ELSE
            s351_tw=eaa_convey(3,3)
          END IF
        END IF
C
C  s-7 new and s-7 spillway capacities
C
        IF (istaopt(is7bsn-6).eq.0) THEN
          nnrc_cap_lower_grav=-901.
          s7_new_hw=-901.
          s7_new_tw=-901.
          s7_spill_tw=s7_tailw
        ELSE
          s7_cap_new=-1.0
          s7_cap=-1.0
          IF (eaa_conv_opt_s7.ne.'PUMP') THEN
            CALL eaaspill (s7_cap_new, eaa_convey(3,4), eaa_convey(3,5),
     &        4, ifacteaa)
            CALL eaaspill (s7_cap, eaa_convey(3,6), eaa_convey(3,7), 2,
     &       ifacteaa)
            s7_new_hw=eaa_convey(3,4)
            s7_new_tw=eaa_convey(3,5)
            s7_spill_hw=eaa_convey(3,6)
            s7_spill_tw=eaa_convey(3,7)
          ELSE
            CALL eaaspill (s7_cap_new, eaa_convey(5,4), eaa_convey(5,5),
     &        4, ifacteaa)
            IF (igravity_s7.eq.1) THEN
              s7_cap=-1.0
              CALL eaaspill (s7_cap, eaa_convey(5,6), eaa_convey(5,7),
     &         2, ifacteaa)
              s7_spill_hw=eaa_convey(5,6)
            ELSE
              s7_spill_hw=pmp_int(is7bsn-6)
              s7_cap=s7max
            END IF
            s7_new_hw=eaa_convey(5,4)
```

```
            s7_new_tw=eaa_convey(5,5)
            s7_spill_tw=s7_tailw
          END IF
        END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C         S-352
C
        s352_tw=eaa_convey(7,3)
C
C         WPB CANAL
C
        fllkwpb=eaa_convey(7,1)
C
C         S-354
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
        IF (istaopt(is8bsn-6).eq.0) THEN
          ireach=1
          dschg_to_meet_s354_dmnd=0.0
          DO iw=1,n_days_week_del_eaa(indx_lok_del)
            IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &       indx_lok_del)) THEN
              dschg_to_meet_s354_dmnd=sum_eaa_dmnd_ssm(is8bsn-6)*
     &         frac_eaa_dmnd_met
            END IF
          END DO
          IF (eaa_conv_opt_s8.ne.'PUMP') THEN
            IF (.not.lforpump(is8bsn-6)) THEN
              istr_eaa=5
            ELSE
              istr_eaa=9
            END IF
            CALL qmax_one_reach (stagelo, s8_tailw, istr_eaa, 1, ireach,
     &        1, grf3amx, s354cap, miamic_cap_upper_grav, s8_cap,
     &       s354_tw, ctw, ifacteaa)
            s8_spill_hw=ctw
            total_neutral_capac=miamic_cap_upper_grav
            s354_tw_neut=s354_tw
            s354cap_neut=s354cap
          ELSE
            IF (.not.lforpump(is8bsn-6)) THEN
              istype=1
              istr_eaa=5
              rmax_cuhw=rmaxstg(is8bsn-6)
              chw=stagelo
              suhw1=stagelo
            ELSE
              istype=2
              istr_eaa=9
              rmax_cuhw=amin1(rmaxstg(is8bsn-6),stagelo+
     &         rmax_lift(is8bsn-6))
              chw=rmax_cuhw
              suhw1=rmax_cuhw
            END IF
            CALL qmax_one_reach (suhw1, s8_tailw, istr_eaa, 1, ireach,
     &       1, grf3amx, s354cap, miamic_cap_upper_grav, s8_cap,
```

```
     &       s354_tw, ctw, ifacteaa)
             IF (igravity_s8_nosta.eq.1) s8_spill_hw=ctw
             delta0=(chw-ctw)/25.0
             ctw=pmp_int(is8bsn-6)
             CALL us_flow (istype, istr_eaa, ireach, 1, chw, ctw,
     &       stagelo, delta0, 1.0, 100, miamic_cap_upper_pump,
     &       s354cap_pump, s354_tw_pump, ifacteaa)
             IF (s354_tw_pump.gt.rmax_cuhw) THEN
               s354_tw_pump=rmax_cuhw
               s354cap_pump=-1.0
               CALL eaaspill (s354cap_pump, stagelo, s354_tw_pump,
     &         istr_eaa, ifacteaa)
               qcanal=-1.0
               IF (s354_tw_pump.gt.ctw) THEN
                 CALL eaa_prof_lu_table (qcanal, s354_tw_pump, ctw,
     &           ireach, ifacteaa)
               ELSE
                 qcanal=0.0
               END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               miamic_cap_upper_pump=qcanal
             END IF
             grf3amx_pump=miamic_cap_upper_pump
             total_neutral_capac=miamic_cap_upper_pump
             s354_tw_neut=s354_tw_pump
             s354cap_neut=s354cap_pump
             IF (igravity_s8_nosta.eq.0) s8_spill_hw=ctw
           END IF
           capac_limit=amax1(total_neutral_capac,cnlcap_ws(imiami))

           dschg_to_eaa=amin1(dschg_to_meet_s354_dmnd,capac_limit)
C            print *,dschg_to_eaa,dschg_to_meet_s354_dmnd,capac_limit
C    +        ,s354cap_neut,total_neutral_capac,LFORPUMP(is8bsn-6)
C    +        ,'   DISCHG TO EAA'
           IF (dschg_to_eaa.gt.s354cap_neut.or.dschg_to_meet_s354_dmnd.gt
     &      .total_neutral_capac) THEN
             IF (.not.lforpump(is8bsn-6)) THEN
               chw=-1.0
               CALL eaaspill (dschg_to_eaa, stagelo, chw, 5, ifacteaa)
               chw=amax1(chw,rmax_tw_eaad(is8bsn-6))
               IF (chw.eq.rmax_tw_eaad(is7bsn-6)) THEN
                 dschg_to_eaa=-1.0
                 CALL eaaspill (dschg_to_eaa, stagelo, chw, 5, ifacteaa)
               END IF
               s354cap=dschg_to_eaa
               s354_tw_neut=chw
               s354_tw=chw
               IF (eaa_conv_opt_s8.eq.'PUMP') s354_tw_pump=chw
               ctw=-1.0
               CALL eaa_prof_lu_table (dschg_to_eaa, s354_tw_neut, ctw,
     &         1, ifacteaa)
               miamic_cap_upper_grav=dschg_to_eaa
               IF (eaa_conv_opt_s8.ne.'PUMP') THEN
                 s8_cap=-1.0
                 CALL eaaspill (s8_cap, ctw, s8_tailw, 1, ifacteaa)
                 s8_spill_hw=s8_tailw
               ELSE
```

```
                s8_cap=s8max
                miamic_cap_upper_pump=dschg_to_eaa
                s8_spill_hw=pmp_int(is8bsn-6)
              END IF
            END IF
          END IF
        ELSE
          IF (eaa_conv_opt_s8.eq.'PUMP') THEN
            s354_tw=eaa_convey(2,3)
          ELSE
            s354_tw=eaa_convey(1,3)
          END IF
        END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C   s-8new and s-8 spillway capacities
C
        IF (istaopt(is8bsn-6).eq.0) THEN
          s8_new_hw=-901.
          s8_new_tw=-901.
          s8_spill_tw=s8_tailw
          miamic_cap_lower_grav=-901.
        ELSE
          s8_cap_new=-1.0
          s8_cap=-1.0
          IF (eaa_conv_opt_s8.ne.'PUMP') THEN
            CALL eaaspill (s8_cap_new, eaa_convey(1,4), eaa_convey(1,5),
     &        3, ifacteaa)
            CALL eaaspill (s8_cap, eaa_convey(1,6), eaa_convey(1,7), 1,
     &       ifacteaa)
            s8_new_hw=eaa_convey(1,4)
            s8_new_tw=eaa_convey(1,5)
            s8_spill_hw=eaa_convey(1,6)
            s8_spill_tw=eaa_convey(1,7)
          ELSE
            CALL eaaspill (s8_cap_new, eaa_convey(2,4), eaa_convey(2,5),
     &        3, ifacteaa)
            IF (igravity_s8.eq.1) THEN
              s8_cap=-1.0
              CALL eaaspill (s8_cap, eaa_convey(2,6), eaa_convey(2,7),
     &         1, ifacteaa)
              s8_spill_hw=eaa_convey(2,6)
            ELSE
              s8_cap=s8max
              s8_spill_hw=pmp_int(is8bsn-6)
            END IF
            s8_new_hw=eaa_convey(2,4)
            s8_new_tw=eaa_convey(2,5)
            s8_spill_tw=s8_tailw
          END IF
        END IF
      END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      s351mx=s351cap
      s352mx=s352cap
      s354mx=s354cap
C
```

```
RETURN
END
```

eaa_neutral_caps.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: eaa_neutral_caps.F,v 1.11 2005/06/10 13:29:03 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaa_neutral_caps.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine calculates conveyance capacities for the major
C   EAA Canals with and without STAs for the neutral case (no EAA
C   runoff/no EAA demand)
C
C  note: upon entry values for s8_tailw and s7_tailw
C        should be passed by COMMON block and look-up table
C        values for s354/s351/s352 gate openings should have
C        been READ-in previously
C
C  List of Reaches
C
C  Base Case
C  1. EAA Miami Canal:           S354HW to S8TW
C  2. EAA North New River Canal: S351HW to S7TW
C  3. EAA Hillsboro Canal:       S351TW to S6HW
C  4. EAA West Palm Beach Canal: S352TW to S5AHW
C
C  Future Base
C  5. EAA Miami Canal:           S354TW to S8newHW
C  6. EAA North New River Canal: S351TW to S7newHW
C  7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
C  8. EAA Miami Canal:           S8newTW to S8HW
C  9. EAA North New River Canal: S7newTW to S7HW
C
C  List of Spillways
C
C    eaa spillway name
C  1.   s8
C  2.   s7
C  3.   s8new
```

```
C  4.   s7new
C  5.   s354             stagelo to lo_str_tw(imiami)
C  6.   s351             stagelo to lo_str_tw(innr)
C  7.   s352             stagelo to lo_str_tw(iwpb)
C  8.   s150
C  9.   s354 forward pump
C  10.  s351 forward pump
C  11.  s352 forward pump
C
C  List of Pumps
C
C    eaa pump name
C  1.   s8pmp
C  2.   s7pmp
C  3.   s6pmp
C  4.   s5apmp
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE eaa_neutral_caps (ifacteaa, pctmet,
     & dmnd_nnr_aft_resasr, res_outflow_to_basin, rlok_stg_beg_forpmp,
     & rlok_stg_end_forpmp, sum_alloc_ssm, sum_losa_dmnd, ssmminfrac,
     & alocnnr_potdel, indx_lok_del, pdsi, frac_dmnd_met_wn,
     & frac_dmnd_met_wt, nbrkpt_ssmwt, nbrkpt_ssmwn, iregjulwt,
     & iregjulwn, regstgwt, regstgwn, sum_eaa_dmnd_ssm_nc)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'eaa_caps.inc'
      INCLUDE 'eaacap.inc'
      INCLUDE 'mgateop.inc'
      INCLUDE 'fin_go.inc'
      INCLUDE 'resws.inc'

      DIMENSION alocnnr_potdel(2), iregjulwn(25), iregjulwt(25),
     & pdsi(12), regstgwn(25), regstgwt(25),
     & res_outflow_to_basin(max_n_res,max_nostr_res),
     & rlok_stg_beg_forpmp(max_n_eaa_condt),
     & rlok_stg_end_forpmp(max_n_eaa_condt),
     & sum_eaa_dmnd_ssm_nc(max_n_eaa_condt)

      REAL iregjulwn, iregjulwt

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C initialize variables

      s6_hw  = pmp_int(is6bsn-6)
```

```
      s5a_hw = pmp_int(is5absn-6)
      IF (itw_opt(is6_no_sim) .EQ. 0) THEN
         s6_tailw = chdep(idn_canal_index(is6_no_sim)) + hdc
     $   (idn_canal_index(is6_no_sim),iseas)
      ELSE
         dpthgw = (ells(node_tw(is6_no_sim)) - h(node_tw(is6_no_sim)))*s
     $   (node_tw(is6_no_sim))
         IF (pond(node_tw(is6_no_sim)) .GT. dpthgw) THEN
            s6_tailw = pond(node_tw(is6_no_sim)) - dpthgw + ells(node_tw
     $      (is6_no_sim))
         ELSE
            s6_tailw = (pond(node_tw(is6_no_sim)) - dpthgw)/s(node_tw
     $      (is6_no_sim)) + ells(node_tw(is6_no_sim))

         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (itw_opt(is150_no_sim) .EQ. 0) THEN
         s150_tailw = chdep(idn_canal_index(is150_no_sim)) + hdc
     $   (idn_canal_index(is150_no_sim),iseas)
      ELSE
         dpthgw = (ells(node_tw(is150_no_sim)) - h(node_tw(is150_no_sim)
     $   ))*s(node_tw(is150_no_sim))
         IF (pond(node_tw(is150_no_sim)) .GT. dpthgw) THEN
            s150_tailw = pond(node_tw(is150_no_sim)) - dpthgw + ells
     $      (node_tw(is150_no_sim))
         ELSE
            s150_tailw = (pond(node_tw(is150_no_sim)) - dpthgw)/s
     $      (node_tw(is150_no_sim)) + ells(node_tw(is150_no_sim))

         ENDIF
      ENDIF
c
C  COMPUTE DEMANDS IN EAA TO BE MET DURING SSM
      QWS1 = agp(is8bsn-6)/86400.
      QWS2 = agp(is7bsn-6)/86400.
      QWS3 = agp(is5absn-6)/86400.
      sum_eaa_dmnd_ssm_nc(is8bsn-6) = sum_eaa_dmnd_ssm_nc
     $       (is8bsn-6) + amax1(QWS1,0.0)
      if (QWS1 .LT. -0.001) sum_eaa_dmnd_ssm_nc(is8bsn-6) = 0.0
      sum_eaa_dmnd_ssm_nc(is7bsn-6) = sum_eaa_dmnd_ssm_nc
     $       (is7bsn-6) + amax1(QWS2,0.0)
      if (QWS2 .LT. -0.001) sum_eaa_dmnd_ssm_nc(is7bsn-6) = 0.0
      sum_eaa_dmnd_ssm_nc(is5absn-6) = sum_eaa_dmnd_ssm_nc
     $       (is5absn-6) + amax1(QWS3,0.0)
      if (QWS3 .LT. -0.001) sum_eaa_dmnd_ssm_nc(is5absn-6) = 0.0
c
c  determine condition of forward pumping - dependent on LOK stage
c
      DO i = 1,3
         IF (stagelo .LE. rlok_stg_beg_forpmp(i) .OR. LFORPUMP(i)) THEN
            LFORPUMP(i) = .TRUE.
            LFORPUMP(i) = .not. (stagelo .GT. rlok_stg_end_forpmp(i))
         ENDIF
      ENDDO
c
c  determine fraction of EAA demand to be met during the days of
```

```
c  delivery from LOK
c
      iday_of_week = idofw(month,iday,ifyr+iyear-1)
      SCH = sched(IDAY,MONTH)
      do idw = 1,n_days_week_del_eaa(indx_lok_del)
         IF (iday_of_week .eq. idays_wk_del_eaa_ssm_indx(idw
     $   ,indx_lok_del)) THEN
            IF (nint(sum_losa_dmnd) .gt. 0) THEN
               frac_eaa_dmnd_met_nc = sum_alloc_ssm / sum_losa_dmnd
               frac_eaa_dmnd_met_nc = amin1(frac_eaa_dmnd_met_nc
     $         ,1.0)
            ELSE
                frac_eaa_dmnd_met_nc = 1.0
            ENDIF
            ssmwatch_l = schedwt(IDAY,MONTH,nbrkpt_ssmwt,iregjulwt
     $       ,regstgwt)
            ssmwarn_l  = schedwn(IDAY,MONTH,nbrkpt_ssmwn,iregjulwn
     $       ,regstgwn)
            rlok_stg_dev_ssm = SCH - stagelo
            if (stagelo .ge. SCH) then
              if (rlok_stg_dev_ssm .gt. pdsi(month)) then
                IF (stagelo .lt. ssmwarn_l ) then
                  frac_eaa_dmnd_met_nc = frac_dmnd_met_wn
                 ELSEIF (stagelo .le. ssmwatch_l) then
                  frac_eaa_dmnd_met_nc = frac_dmnd_met_wt
                ELSE
                  frac_eaa_dmnd_met_nc = 1.0
                ENDIF
              else
                frac_eaa_dmnd_met_nc = 1.0
              endif
            endif
         ENDIF
         frac_eaa_dmnd_met_nc=amax1(ssmminfrac,frac_eaa_dmnd_met_nc)
      enddo

c tolerq = convergence criterion for matching flows
      tolerq = 5.
c itermax = maximum allowable number of iterations
      itermax = 100
       dschg_to_meet_eaa_dmnd = 0.0
       DO IW = 1,n_days_week_del_eaa(indx_lok_del)
          IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $    ,indx_lok_del)) THEN
             dschg_to_meet_eaa_dmnd = sum_eaa_dmnd_ssm_nc(is8bsn-6)
     $       * frac_eaa_dmnd_met_nc
             sum_eaa_dmnd_ssm_nc(is8bsn-6) = 0.0
          ENDIF
       ENDDO
c     dschg_to_meet_eaa_dmnd = amax1(agp(is8bsn-6),0.0) * pctmet/86400.
c
c
c for miami canal (gravity flow)
      IF (itw_opt(is8_no_sim) .EQ. 0) THEN
         s8_tailw = chdep(idn_canal_index(is8_no_sim)) + hdc
     $   (idn_canal_index(is8_no_sim),iseas)
      ELSE
```

```
         dpthgw = (ells(node_tw(is8_no_sim)) - h(node_tw(is8_no_sim)))*s
     $   (node_tw(is8_no_sim))
         IF (pond(node_tw(is8_no_sim)) .GT. dpthgw) THEN
            s8_tailw = pond(node_tw(is8_no_sim)) - dpthgw + ells(node_tw
     $      (is8_no_sim))
         ELSE
            s8_tailw = (pond(node_tw(is8_no_sim)) - dpthgw)/s(node_tw
     $      (is8_no_sim)) + ells(node_tw(is8_no_sim))

         ENDIF
      ENDIF
c
      IF (istaopt(is8bsn-6) .EQ. 1) THEN
         iagbsn = 1
         add_conv_red_miami = 0.0
         DO irs = 1,nres_ws
            IF (iconv_use_opt_ws(irs) .EQ. 1) THEN
               DO js = 1,noutlets(irs)
                  IF (ieaa_bsn(irs,js) .EQ. is8bsn-6) THEN
                     add_conv_red_miami = add_conv_red_miami +
     $               res_outflow_to_basin(irs,js)
                  ENDIF
               END DO
            ENDIF
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (.not. LFORPUMP(is8bsn-6)) THEN
         IF (stagelo.LE.s8_tailw) THEN
            qconvey = 0.0
            qcanal  = 0.0
            s8spmx  = 0.0
            dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,cnlcap_ws
     $      (imiami))
            IF (dschg_to_eaa .GT. 0.0) THEN
               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               ENDIF
               total_ws_miami = dschg_to_eaa + add_conv_red_miami
            ELSE
               total_ws_miami = add_conv_red_miami
               cuhw = stagelo
            ENDIF
            IF (nint(total_ws_miami) .GT. 0) THEN
               cutw = -1.0
               call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $         ifacteaa)
            ELSE
               cutw = cuhw
            ENDIF
            miamic_cap_upper_grav = total_ws_miami
            miamic_cap_lower_grav = 0.0
            clhw = cutw
            cltw = cutw
```

```
            s354_tw = cuhw
            cuhwfinal = cuhw
            s354cap = dschg_to_eaa
c calculate actual gate opening
            IF (stagelo .GT. cuhwfinal) THEN
               gateop1 = dschg_to_eaa/(276.0*sqrt(stagelo-cuhwfinal))
               IF (ifacteaa .EQ. 1) THEN
                  gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
               ENDIF
            ELSE
               gateop1 = 0.0
            ENDIF
            IF (gateop1 .LE. 8.3) THEN
               go_s354 = gateop1
            ELSE
               go_s354 = 8.3
            ENDIF
            goto 30
         ENDIF
         IF (stagelo .GT. rmaxstg(imiami)) THEN
            cuhw = rmaxstg(imiami)
            s354cap = -1.0
            call eaaspill(s354cap,stagelo,cuhw,5,ifacteaa)
         ELSE
            cuhw = stagelo
            s354cap = 0.0
         ENDIF
         sltw = s8_tailw
         IF (cuhw .GT. s8_tailw) THEN
            cltw = s8_tailw
         ELSE
            qconvey = 0.0
            qcanal  = 0.0
            s8spmx  = 0.0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,cnlcap_ws
     $      (imiami))
            IF (dschg_to_eaa .GT. 0.0) THEN
               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               ENDIF
               total_ws_miami = dschg_to_eaa + add_conv_red_miami
            ELSE
               total_ws_miami = add_conv_red_miami
               cuhw = stagelo
            ENDIF
            IF (nint(total_ws_miami) .GT. 0) THEN
               cutw = -1.0
               call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $         ifacteaa)
            ELSE
               cutw = cuhw
            ENDIF
```

```
            miamic_cap_upper_grav = total_ws_miami
            miamic_cap_lower_grav = 0.0
            clhw = cutw
            cltw = cutw
            s354_tw = cuhw
            s354cap = dschg_to_eaa
c calculate actual gate opening
            IF (stagelo .GT. cuhwfinal) THEN
               gateop1 = dschg_to_eaa/(276.0*sqrt(stagelo-cuhwfinal))
               IF (ifacteaa .EQ. 1) THEN
                  gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
               ENDIF
            ELSE
               gateop1 = 0.0
            ENDIF
            IF (gateop1 .LE. 8.3) THEN
               go_s354 = gateop1
            ELSE
               go_s354 = 8.3
            ENDIF
            goto 30
         ENDIF
         istype = 1
         istrm = 3
         istrl = 1
         ireachu = 5
         ireachl = 8
         delta0 = (cuhw-cltw)/25.
         delta1 = delta0
c routine dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
c                   cltw,sltw,delta0,toler,canalq,qlstrr,slhw)
         call dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
     $   cltw,cutw,clhw,sltw,delta0,tolerq,itermax,qcanal,s8spmx,slhw,
     $   ifacteaa)
         IF (s8spmx.LE.s354cap) THEN
            qconvey = s8spmx
c       OR  qconvey = qcanal
c Actually, after dsds_flow is called,
c           qcanal ==  qucanal = qmstr = qlcanal = qlstrr(=s8spmx)
c calculate actual gate opening
            IF (stagelo .GT. cuhw) THEN
               gateop1 = s8spmx / (276.0*sqrt(stagelo-cuhw))
            ELSE
               gateop1 = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            conv_cap_ws = amax1(cnlcap_ws(imiami) ,qcanal
     $      -add_conv_red_miami)
            dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,conv_cap_ws)

            IF (dschg_to_eaa .GT. s354cap .OR. dschg_to_meet_eaa_dmnd
     $      .GT. amax1(qcanal-add_conv_red_miami ,0.0)) THEN

               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
```

```
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               ENDIF
               s354cap = dschg_to_eaa
               cutw = -1.0
               total_ws_miami = dschg_to_eaa + add_conv_red_miami
               call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $         ifacteaa)
               miamic_cap_upper_grav = total_ws_miami
               call qmax_one_reach(cutw,s8_tailw,3,1,8,iagbsn ,
     $         qconvey_lower,s8_cap_new,miamic_cap_lower_grav ,s8_cap,
     $         clhw,cltw,ifacteaa)
c calculate actual gate opening
               IF (stagelo .GT. cuhw) THEN
                  gateop1 = dschg_to_eaa /(276.0*sqrt(stagelo-cuhw))
                  IF (ifacteaa .EQ. 1) THEN
                     gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
                  ENDIF
               ELSE
                  gateop1 = 0.0
               ENDIF
            ELSE
               miamic_cap_upper_grav = amax1(qcanal,0.0)
               miamic_cap_lower_grav = amax1(qcanal,0.0)
            ENDIF
            IF (gateop1 .LE. 8.3) THEN
               go_s354 = gateop1
            ELSE
               go_s354 = 8.3
            ENDIF
c
            goto 30
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         iter = 0
   10    cltw = cltw - delta1
         s8spmx = -1.0
         call eaaspill(s8spmx,cltw,s8_tailw,1,ifacteaa)
         qlcanal = s8spmx
         ireachu = 5
         istrm = 3
         ireachl = 8
         cuhw = -1.0
         ireach = 5
         call eaa_prof_two_reaches(qlcanal,cuhw,cltw,cutw,clhw,ireachu,
     $   istrm,ireachl,iagbsn,itermax,ifacteaa)
         s354cap = -1.0
         call eaaspill(s354cap,stagelo,cuhw,5,ifacteaa)
c
         iter = iter + 1
         IF (iter .EQ. itermax)goto 20
c
         IF (s8spmx-s354cap .GT. tolerq) THEN
c CONTINUE to decrease cltw such that s354cap increases and s8spmx decreases
            goto 10
         ENDIF
         IF (s354cap-s8spmx .GT. tolerq) THEN
```

```
c overshot estimation of new cltw; redo with 1/2 delta1
            cltw = cltw + delta1
            delta1 = delta1/2.
            goto 10
         ENDIF
   20    qcanal = qlcanal
         conv_cap_ws = amax1(cnlcap_ws(imiami) ,qcanal
     $   -add_conv_red_miami)
         dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd,conv_cap_ws)
         IF (dschg_to_eaa .GT. s354cap .OR. dschg_to_meet_eaa_dmnd .GT.
     $   amax1(qcanal-add_conv_red_miami ,0.0)) THEN

            cuhw = -1.0
            call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
            cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
            IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
               dschg_to_eaa = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
            ENDIF
            s354cap = dschg_to_eaa
            cutw = -1.0
            call eaa_prof_lu_table(dschg_to_eaa,cuhw ,cutw,5,ifacteaa)

            miamic_cap_upper_grav = dschg_to_eaa
            call qmax_one_reach(cutw,s8_tailw,3,1,8,iagbsn ,
     $      qconvey_lower,s8_cap_new,miamic_cap_lower_grav ,s8_cap,clhw,
     $      cltw,ifacteaa)
c calculate actual gate opening
            IF (stagelo .GT. cuhw) THEN
               gateop1 = dschg_to_eaa /(276.0*sqrt(stagelo-cuhw))
               IF (ifacteaa .EQ. 1) THEN
                  gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
               ENDIF
            ELSE
               gateop1 = 0.0
            ENDIF
            IF (gateop1 .LE. 8.3) THEN
               go_s354 = gateop1
            ELSE
               go_s354 = 8.3
            ENDIF
         ELSE
            miamic_cap_upper_grav = amax1(qcanal,0.0)
            miamic_cap_lower_grav = amax1(qcanal,0.0)
         ENDIF
         ELSE
c  forward pumping when LOK is low
            istype = 1
            istrm = 3
            istrl = 1
            ireachu = 5
            ireachl = 8
            rmax_cuhw = amin1(stagelo+rmax_lift(is8bsn-6)
     $      ,rmaxstg(is8bsn-6))
            cuhw = rmax_cuhw
            sltw = s8_tailw
            cltw = sltw
```

```
            s354cap = -1.0
            call eaaspill (s354cap,stagelo,cuhw,9,ifacteaa)
            qcanal = s354cap
            s8spmx = s354cap
            cltw = -1.0
            call eaaspill (s8spmx,cltw,s8_tailw,1
     $      ,ifacteaa)
            cuhw = -1.0
            call eaa_prof_two_reaches (qcanal,cuhw,cltw,cutw,clhw
     $      ,ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
            IF (cuhw .gt. rmax_cuhw) THEN
               cuhw = rmax_cuhw
               delta0 = (cuhw-cltw)/25.
               delta1 = delta0
               call dsds_flow(istype,istrm,istrl,ireachu,ireachl
     $         ,iagbsn,cuhw,cltw,cutw,clhw,sltw,delta0,tolerq
     $         ,itermax,qcanal,s8spmx,slhw,ifacteaa)
            ENDIF
            miamic_cap_upper_grav = amax1(qcanal,0.0)
            miamic_cap_lower_grav = amax1(qcanal,0.0)
         ENDIF

c adjustment within tolerance; exit
   30    qcanal = amax1(qcanal,0.0)
         qconvey = amin1(s354cap,qcanal,s8spmx)
         eaa_convey(1,1) = qconvey
         eaa_convey(1,2) = stagelo
         eaa_convey(1,3) = cuhw
         eaa_convey(1,4) = cutw
         eaa_convey(1,5) = clhw
         eaa_convey(1,6) = cltw
         eaa_convey(1,7) = s8_tailw
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c  when simulating ability to pump water supply to WCAS and LEC
c  check IF headwater at existing s8 spillway is greater THEN
c  intake level of pump. If greater,use the pump;otherwise
c  use the spillway.
c
         IF (eaa_conv_opt_s8 .EQ. 'PUMP') THEN
            IF (eaa_convey(1,6) .LE. pmp_int(is8bsn-6)) THEN
               DO i = 1,7
                  eaa_convey(2,i) = eaa_convey(1,i)
               END DO
               miamic_cap_upper_pump = miamic_cap_upper_grav
               miamic_cap_lower_pump = miamic_cap_lower_grav
               igravity_s8 = 1
               GO TO 50
            ELSE
               igravity_s8 = 0
            ENDIF
         ENDIF
c
c>>>>>>>> PROFILE 2 (proc2)
c         READ(10,'(2i9,3f9.0)',end=123)
c     $      iday,month,stagelo,chdep(1),s8pmphw,chdep(2)
c note: s8pmphw is used as structure headwater for s-8 in this context
c
```

```
c for miami canal (pumped flow)
         IF (eaa_conv_opt_s8 .EQ. 'PUMP') THEN
            iagbsn = 1
            IF (.not. LFORPUMP(is8bsn-6)) THEN
            IF (stagelo .LE. pmp_int(is8bsn-6)) THEN
               qconvey = 0.0
               qcanal = 0.0
               s8pmpmx = 0.0
               dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,cnlcap_ws
     $         (imiami))
               IF (dschg_to_eaa .GT. 0.0) THEN
                  cuhw = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
                  cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
                  IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
                     dschg_to_eaa = -1.0
                     call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
                  ENDIF
                  total_ws_miami = dschg_to_eaa + add_conv_red_miami
               ELSE
                  total_ws_miami = add_conv_red_miami
                  cuhw = stagelo
               ENDIF
               IF (nint(total_ws_miami) .GT. 0) THEN
                  cutw = -1.0
                  call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $            ifacteaa)
               ELSE
                  cutw = cuhw
               ENDIF
               miamic_cap_upper_pump = total_ws_miami
               miamic_cap_lower_pump = 0.0
               clhw = cutw
               cltw = cutw
               s354_tw = cuhw
               cuhwfinal = cuhw
               s354cap = dschg_to_eaa
c calculate actual gate opening
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               IF (stagelo .GT. cuhwfinal) THEN
                  gateop1 = dschg_to_eaa/(276.0*sqrt(stagelo-cuhwfinal))
                  IF (ifacteaa .EQ. 1) THEN
                     gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
                  ENDIF
               ELSE
                  gateop1 = 0.0
               ENDIF
               IF (gateop1 .LE. 8.3) THEN
                  go_s354 = gateop1
               ELSE
                  go_s354 = 8.3
               ENDIF
               goto 40
            ENDIF
            IF (stagelo .GT. rmaxstg(imiami)) THEN
               cuhw = rmaxstg(imiami)
               s354cap = -1.0
```

```
               call eaaspill(s354cap,stagelo,cuhw,5,ifacteaa)
            ELSE
               cuhw = stagelo
               s354cap = 0.0
            ENDIF
            sltw = s8_tailw
            IF (cuhw .GT. pmp_int(imiami)) THEN
               cltw = pmp_int(imiami)
            ELSE
               qconvey = 0.0
               qcanal = 0.0
               s8pmpmx = 99999.99
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,cnlcap_ws
     $         (imiami))
               IF (dschg_to_eaa .GT. 0.0) THEN
                  cuhw = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
                  cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
                  IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
                     dschg_to_eaa = -1.0
                     call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
                  ENDIF
                  total_ws_miami = dschg_to_eaa + add_conv_red_miami
               ELSE
                  total_ws_miami = add_conv_red_miami
                  cuhw = stagelo
               ENDIF
               IF (nint(total_ws_miami) .GT. 0) THEN
                  cutw = -1.0
                  call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $            ifacteaa)
               ELSE
                  cutw = cuhw
               ENDIF
               miamic_cap_upper_pump = total_ws_miami
               miamic_cap_lower_pump = 0.0
               clhw = cutw
               cltw = cutw
               s354_tw = cuhw
               cuhwfinal = cuhw
               s354cap = dschg_to_eaa
c calculate actual gate opening
               IF (stagelo .GT. cuhwfinal) THEN
                  gateop1 = dschg_to_eaa/(276.0*sqrt(stagelo-cuhwfinal))
                  IF (ifacteaa .EQ. 1) THEN
                     gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
                  ENDIF
               ELSE
                  gateop1 = 0.0
               ENDIF
               IF (gateop1 .LE. 8.3) THEN
                  go_s354 = gateop1
               ELSE
                  go_s354 = 8.3
               ENDIF
```

```
               goto 40
            ENDIF
            istru = 5
            istype = 1
            istrm = 3
            istrl = 1
            ireachu = 5
            ireachl = 8
            delta0 = (cuhw-cltw)/25.
            cuhwmax = cuhw
c
c routine usus_flow(istype,istrm,istru,ireachu,ireachl,iagbsn,
c     $  cuhw,cltw,cutw,clhw,suhw,delta0,toler,itermax,canalq,struq,sutw)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            call usus_flow(istype,istrm,istru,ireachu,ireachl,iagbsn,
     $      cuhw,cltw,cutw,clhw,stagelo,delta0,tolerq, itermax,canalq,
     $      s354q,sutw,ifacteaa)
            IF (cuhw .GE. cuhwmax) THEN
c s354 tw limit is exceeded by optimum stage
               cuhwfinal = cuhwmax
               s354cap = -1.0
               call eaaspill(s354cap,stagelo,cuhwfinal,5,ifacteaa)
               qcanal = -1.0
               call eaa_prof_two_reaches(qcanal,cuhwfinal,cltw,cutw,
     $         clhw, ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
            ELSE
               cuhwfinal = cuhw
               s354cap = s354q
               qcanal = canalq
            ENDIF
            s8pmpmx = 99999.99
            qcanal = amax1(qcanal,0.0)
c
            qconvey = amin1(s354cap,qcanal,s8pmpmx)
c calculate actual gate opening
            IF (stagelo .GT. cuhwfinal) THEN
               gateop1 = qconvey /(276.0*sqrt(stagelo-cuhwfinal))
               IF (ifacteaa .EQ. 1) THEN
                  gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
               ENDIF
            ELSE
               gateop1 = 0.0
            ENDIF
c
            IF (gateop1 .LE. 8.3) THEN
               go_s354 = gateop1
            ELSE
               go_s354 = 8.3
            ENDIF
            conv_cap_ws = amax1(cnlcap_ws(imiami),qcanal
     $      -add_conv_red_miami)
            dschg_to_eaa = amin1(dschg_to_meet_eaa_dmnd ,conv_cap_ws)

            IF (dschg_to_eaa .GT. s354cap .OR. dschg_to_meet_eaa_dmnd
     $      .GT. amax1(qcanal-add_conv_red_miami ,0.0)) THEN

               cuhw = -1.0
```

```
               call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is8bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is8bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,5,ifacteaa)
               ENDIF
               cuhwfinal = cuhw
               total_ws_miami = dschg_to_eaa + add_conv_red_miami
               cutw = -1.0
               call eaa_prof_lu_table(total_ws_miami,cuhw ,cutw,5,
     $         ifacteaa)
               miamic_cap_upper_pump = total_ws_miami
               chw = cutw
               ctw = pmp_int(is8bsn-6)
               delta0 = (chw-ctw)/25.
               call us_flow(1,3,8,iagbsn,chw,ctw,cutw,delta0 ,1.0,100,
     $         miamic_cap_lower_pump,s8_cap_new ,clhw,ifacteaa)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c calculate actual gate opening
               IF (stagelo .GT. cuhwfinal) THEN
                  gateop1 = dschg_to_eaa/ (276.0*sqrt(stagelo-cuhwfinal)
     $            )
                  IF (ifacteaa .EQ. 1) THEN
                     gateop1 = gateop1 / rcpfacteaa(is8bsn-6)
                  ENDIF
               ELSE
                  gateop1 = 0.0
               ENDIF
               IF (gateop1 .LE. 8.3) THEN
                  go_s354 = gateop1
               ELSE
                  go_s354 = 8.3
               ENDIF
               s354cap = dschg_to_eaa
            ELSE
               miamic_cap_upper_pump = amax1(qcanal,0.0)
               miamic_cap_lower_pump = amax1(qcanal,0.0)
            ENDIF
c
   40       CONTINUE
            ELSE
c  forward pumping when LOK stage is low
               ireachu = 5
               ireachl = 8
               rmax_cuhw = amin1(stagelo+rmax_lift(is8bsn-6)
     $         ,rmaxstg(is8bsn-6))
               cuhw = rmax_cuhw
               s354cap = -1.0
               call eaaspill (s354cap,stagelo,cuhw,9,ifacteaa)
               qcanal = s354cap
               sltw = s8_tailw
               cltw = pmp_int(imiami)
               cuhwfinal = -1.0
               call eaa_prof_two_reaches(qcanal,cuhwfinal,cltw,cutw,
     $         clhw, ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
               IF (cuhwfinal .GT. rmax_cuhw) then
                  cuhwfinal = rmax_cuhw
```

```
                  qcanal = -1.0
                  call eaa_prof_two_reaches(qcanal,cuhwfinal,cltw,
     $            cutw,clhw, ireachu,istrm,ireachl,iagbsn,
     $            itermax,ifacteaa)
               ENDIF
               miamic_cap_upper_pump = amax1(qcanal,0.0)
               miamic_cap_lower_pump = amax1(qcanal,0.0)
            ENDIF
            eaa_convey(2,1) = qconvey
            eaa_convey(2,2) = stagelo
            eaa_convey(2,3) = cuhwfinal
            eaa_convey(2,4) = cutw
            eaa_convey(2,5) = clhw
            eaa_convey(2,6) = cltw
         ENDIF
      ENDIF
c
c>>>>>>>> PROFILES 3 + 4 (proc4g)
c
c for north new river (gravity flow) and hillsboro (pumped flow) canals
c
   50 IF (itw_opt(is7_no_sim) .EQ. 0) THEN
         s7_tailw = chdep(idn_canal_index(is7_no_sim)) + hdc
     $   (idn_canal_index(is7_no_sim),iseas)
      ELSE
         dpthgw = (ells(node_tw(is7_no_sim)) - h(node_tw(is7_no_sim)))*s
     $   (node_tw(is7_no_sim))
         IF (pond(node_tw(is7_no_sim)) .GT. dpthgw) THEN
            s7_tailw = pond(node_tw(is7_no_sim)) - dpthgw + ells(node_tw
     $      (is7_no_sim))
         ELSE
            s7_tailw = (pond(node_tw(is7_no_sim)) - dpthgw)/s(node_tw
     $      (is7_no_sim)) + ells(node_tw(is7_no_sim))

         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (istaopt(is7bsn-6) .EQ. 1) THEN
         iagbsn = 2
         add_conv_red_nnrhl = 0.0
         DO irs = 1,nres_ws
            IF (iconv_use_opt_ws(irs) .EQ. 1) THEN
               DO js = 1,noutlets(irs)
                  IF (ieaa_bsn(irs,js) .EQ. is7bsn-6) THEN
                     add_conv_red_nnrhl = add_conv_red_nnrhl +
     $               res_outflow_to_basin(irs,js)
                  ENDIF
               END DO
            ENDIF
         END DO
         dschg_to_meet_s351_dmnd = 0.0
         DO IW = 1,n_days_week_del_eaa(indx_lok_del)
            IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $      ,indx_lok_del)) THEN
               dschg_to_meet_s351_dmnd= sum_eaa_dmnd_ssm_nc(is7bsn-6)
     $         * frac_eaa_dmnd_met_nc
            ENDIF
```

```
         ENDDO
c        dschg_to_meet_s351_dmnd= amax1(agp(is7bsn-6),0.0)*pctmet/86400.
         dmnd_nnr = amax1(alocnnr_potdel(indx_lok_del),0.0)
     $   * frac_eaa_dmnd_met_nc
         IF (.not. LFORPUMP(is7bsn-6)) THEN
         IF (stagelo .LE. s7_tailw) THEN
            qconvey_nnrc = 0.0
            qconvey_hlsb = 0.0
            qcanal_nnrc  = 0.0
            s7spmx  = 0.0
            s6pmpmx  = 99999.99
            dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,cnlcap_ws
     $      (innr) + cnlcap_ws(ihill))
c
            IF (dschg_to_eaa .GT. 0.0) THEN
               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               ENDIF
            ELSE
               cuhw = stagelo
            ENDIF
            IF (dschg_to_meet_s351_dmnd .GT. 0.0) THEN
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
            ELSE
               ratio_ws = 0.0
            ENDIF
            supply_to_nnrc = ratio_ws*amax1(alocnnr_potdel(indx_lok_del)
     $        ,0.0) * frac_eaa_dmnd_met_nc
            rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $      supply_to_nnrc,0.0)
            rem_pot_supply_excess_hillcap =amax1
     $      (rem_pot_supply_to_nnrhl - cnlcap_ws(ihill),0.0)
            dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            IF (nint(total_nnrc_supply) .GT. 0) THEN
               cutw = -1.0
               call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $         ifacteaa)
            ELSE
               cutw = cuhw
            ENDIF
            nnrc_cap_upper_grav = total_nnrc_supply
            nnrc_cap_lower_grav = 0.0
            clhw = cutw
            cltw = cutw
            s351_tw = cuhw
            s351cap = dschg_to_eaa
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            clhw  = cutw
            cltw  = cutw
c
c  hillsboro canal
```

```
c
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,0.0)
     $         /86400.
            ELSE
               dmnd_hill = 0.0
            ENDIF
            dmnd_hill = amax1(dmnd_hill,0.0)
            supp_to_hill_frm_res = add_conv_red_nnrhl - dmnd_nnr_orig +
     $      dmnd_nnr_aft_resasr
            supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
            total_supp_to_hill = supp_to_hill_frm_res + dmnd_hill
            IF (total_supp_to_hill .GT. 0.0) THEN
               ctw = -1.0
               call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $         ifacteaa)
            ELSE
               ctw = cuhw
            ENDIF
            chw = cuhw
            goto 80
         ENDIF
         IF (stagelo .GT. rmaxstg(innr)) THEN
            cuhw = rmaxstg(innr)
         ELSE
            cuhw = stagelo
            s351cap = 0.0
         ENDIF
         chw = cuhw
         sltw = s7_tailw
         IF (cuhw .GT. s7_tailw) THEN
            cltw = s7_tailw
         ELSE
            qconvey_nnrc = 0.0
            qconvey_hlsb = 0.0
            qcanal_nnrc  = 0.0
            s7spmx  = 0.0
            s6pmpmx  = 99999.99
c
            dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,cnlcap_ws
     $      (innr) + cnlcap_ws(ihill))
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (dschg_to_eaa .GT. 0.0) THEN
               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
               cuhw = amin1(cuhw,rmaxstg(is7bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               ENDIF
            ELSE
               cuhw = stagelo
            ENDIF
            cutw = -1.0
            IF (dschg_to_meet_s351_dmnd .GT. 0.0) THEN
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
```

```
            ELSE
               ratio_ws = 0.0
            ENDIF
            supply_to_nnrc = ratio_ws*amax1(alocnnr_potdel(indx_lok_del)
     $       ,0.0) * frac_eaa_dmnd_met_nc
            rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $      supply_to_nnrc,0.0)
            rem_pot_supply_excess_hillcap =amax1
     $      (rem_pot_supply_to_nnrhl - cnlcap_ws(ihill),0.0)
            dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            IF (nint(total_nnrc_supply) .GT. 0) THEN
               cutw = -1.0
               call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $         ifacteaa)
            ELSE
               cutw = cuhw
            ENDIF
            nnrc_cap_upper_grav = total_nnrc_supply
            nnrc_cap_lower_grav = 0.0
            clhw = cutw
            cltw = cutw
            s351_tw = cuhw
            s351cap = dschg_to_eaa
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            clhw  = cutw
            cltw  = cutw
c
c  hillsboro canal
c
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,0.0)
     $         /86400.
            ELSE
               dmnd_hill = 0.0
            ENDIF
            dmnd_hill = amax1(dmnd_hill,0.0)
            supp_to_hill_frm_res = add_conv_red_nnrhl - dmnd_nnr_orig +
     $      dmnd_nnr_aft_resasr
            supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
            total_supp_to_hill = supp_to_hill_frm_res + dmnd_hill
            IF (total_supp_to_hill .GT. 0.0) THEN
               ctw = -1.0
               call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $         ifacteaa)
            ELSE
               ctw = cuhw
            ENDIF
            chw = cuhw
            goto 80
         ENDIF
         ctw = pmp_int(ihill)
         istype = 1
         istrm = 4
         istrl = 2
         istr_eaa = 6
```

```
         ireachu = 6
         ireachl = 9
         ireach = 7
         delta0 = (cuhw-cltw)/25.
         delta1 = delta0
c
         s351cap = -1.0
         call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
c routine dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
c                   cltw,sltw,delta0,toler,canalq,strq,slhw)
         call dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
     $   cltw,cutw,clhw,sltw,delta0,tolerq,itermax, qcanal_nnrc,s7spmx,
     $   slhw,ifacteaa)
         qcanal_nnrc = amax1(qcanal_nnrc,0.0)
         qconvey_nnrc = amin1(qcanal_nnrc,s7spmx)
c use the same nnr canal headwater for hlsb
         chw = cuhw
         qconvey_hlsb = -1.0
         IF (chw .GT. ctw) THEN
            call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,ifacteaa)
         ELSE
            qconvey_hlsb = 0.0
         ENDIF
         qconvey_hlsb = amax1(qconvey_hlsb,0.0)
         qconvey_total = qconvey_nnrc + qconvey_hlsb
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (s351cap .GE. qconvey_total) goto 70
c
c  decrease cuhw (=s351tw), thus increasing s351cap and decreasing
c  both qconvey_nnrc and qconvey_hlsb
         iter = 0
   60    cuhw = cuhw - delta1
         s351cap = -1.0
         call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
         call dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
     $   cltw,cutw,clhw,sltw,delta0,tolerq,itermax, qcanal_nnrc,s7spmx,
     $   slhw,ifacteaa)
         qconvey_nnrc = amin1(qcanal_nnrc,s7spmx)
c use the same nnr canal headwater for hlsb
         chw = cuhw
         qconvey_hlsb = -1.0
         IF (chw .GT. ctw) THEN
            call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,ifacteaa)
         ELSE
            qconvey_hlsb = 0.0
         ENDIF
         qconvey_total = qconvey_nnrc + qconvey_hlsb
c
         iter = iter + 1
         IF (iter .EQ. itermax)goto 70
c
         IF (qconvey_total-s351cap .GT. tolerq/5.) THEN
c CONTINUE to decrease cuhw such that s351cap increases and qconvey_total decrea
            goto 60
         ENDIF
```

```
         IF (s351cap-qconvey_total .GT. tolerq/5.) THEN
c overshot estimation of new cuhw; redo with 1/2 delta1
            cuhw = cuhw + delta1
            delta1 = delta1/2.
            goto 60
         ENDIF
c adjustment within tolerance; exit
   70    dschg_to_meet_s351_dmnd = 0.0
         DO IW = 1,n_days_week_del_eaa(indx_lok_del)
            IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $      ,indx_lok_del)) THEN
               dschg_to_meet_s351_dmnd= sum_eaa_dmnd_ssm_nc(is7bsn-6)
     $         * frac_eaa_dmnd_met_nc
            ENDIF
         ENDDO
c  70    dschg_to_meet_s351_dmnd= amax1(agp(is7bsn-6),0.0)*pctmet/86400.
c
         conv_max_ws = amax1(cnlcap_ws(innr) + cnlcap_ws(ihill) ,
     $   qconvey_total-add_conv_red_nnrhl)
         dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,conv_max_ws)

         IF (dschg_to_eaa .GT. s351cap .OR. dschg_to_meet_s351_dmnd .GT.
     $   amax1(qconvey_total-add_conv_red_nnrhl,0.0)) THEN
            cuhw = -1.0
            call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
            cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
            IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
               dschg_to_eaa = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
            ENDIF
            s351cap = dschg_to_eaa
            cutw = -1.0
            ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
            supply_to_nnrc = ratio_ws*amax1(alocnnr_potdel(indx_lok_del)
     $      ,0.0) * frac_eaa_dmnd_met_nc
            rem_pot_supply_to_nnrhl = dschg_to_eaa - supply_to_nnrc
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            add_conv_red_hill = add_conv_red_nnrhl - dmnd_nnr_orig +
     $      dmnd_nnr_aft_resasr
            rmax_cap_ws = amax1(cnlcap_ws(ihill),qconvey_hlsb -
     $      add_conv_red_hill)
            rem_pot_supply_excess_hillcap =amax1
     $      (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
            dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $      ifacteaa)
            nnrc_cap_upper_grav = total_nnrc_supply
            call qmax_one_reach(cutw,s7_tailw,4,2,9,iagbsn ,
     $      qconvey_lower,s7_cap_new,nnrc_cap_lower_grav ,s7_cap,clhw,
     $      cltw,ifacteaa)
c
c  hillsboro canal
c
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,0.0)
```

```
     $         /86400.
            ELSE
               dmnd_hill = 0.0
            ENDIF
            dmnd_hill = amax1(dmnd_hill,0.0)
            supp_to_hill_frm_res = add_conv_red_nnrhl - dmnd_nnr_orig +
     $      dmnd_nnr_aft_resasr
            supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
            dischg_to_hill = dmnd_hill
            total_supp_to_hill = supp_to_hill_frm_res + dischg_to_hill
            ctw = -1.0
            call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $      ifacteaa)
            s6_hw = ctw
            chw = cuhw
         ELSE
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del
     $         ),0.0) * frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $         supply_to_nnrc,0.0)
               add_conv_red_hill = add_conv_red_nnrhl - dmnd_nnr_orig +
     $         dmnd_nnr_aft_resasr
               rmax_cap_ws = amax1(cnlcap_ws(ihill),qconvey_hlsb -
     $         add_conv_red_hill)
               rem_pot_supply_excess_hillcap=amax1
     $         (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
               dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            ELSE
               dmnd_nnr = 0.0
            ENDIF
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            nnrc_cap_upper_grav = amax1(qconvey_nnrc ,total_nnrc_supply)
         ENDIF
         ELSE
c  forward pumping when LOK stage is low
            ctw = pmp_int(ihill)
            sltw = s7_tailw
            cltw = s7_tailw
            istype = 1
            istrm = 4
            istrl = 2
            istr_eaa = 10
            ireachu = 6
            ireachl = 9
            ireach = 7
            rmax_cuhw = amin1(stagelo+rmax_lift(is7bsn-6)
     $      ,rmaxstg(is7bsn-6))
            cuhw = rmax_cuhw
            delta0 = (cuhw-cltw)/25.
            delta1 = delta0
c
            iter = 0
   75       cuhw = cuhw - delta1
            s351cap = -1.0
```

```
            call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
c routine dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,cuhw,
c                   cltw,sltw,delta0,toler,canalq,strq,slhw)
c
            call dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,
     $      cuhw,cltw,cutw,clhw,sltw,delta0,tolerq,itermax,
     $      qcanal_nnrc,s7spmx,slhw,ifacteaa)
            qconvey_nnrc = amin1(qcanal_nnrc,s7spmx)
c use the same nnr canal headwater for hlsb
            chw = cuhw
            qconvey_hlsb = -1.0
            IF (chw .GT. ctw) THEN
               call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach
     $         ,ifacteaa)
            ELSE
               qconvey_hlsb = 0.0
            ENDIF
            qconvey_total = qconvey_nnrc + qconvey_hlsb
c
            iter = iter + 1
            IF (iter .EQ. itermax)goto 78
c
            IF (qconvey_total-s351cap .GT. tolerq/5.) THEN
c CONTINUE to decrease cuhw such that s351cap increases and qconvey_total decrea
               goto 75
            ENDIF
            IF (s351cap-qconvey_total .GT. tolerq/5.) THEN
c overshot estimation of new cuhw; redo with 1/2 delta1
               cuhw = cuhw + delta1
               delta1 = delta1/2.
               goto 75
            ENDIF
   78       IF (cuhw .gt. rmax_cuhw) THEN
               cuhw = rmax_cuhw
               call dsds_flow(istype,istrm,istrl,ireachu,ireachl,
     $         iagbsn,cuhw,cltw,cutw,clhw,sltw,delta0,tolerq,
     $         itermax,qcanal_nnrc,s7spmx,slhw,ifacteaa)
               qconvey_nnrc = amin1(qcanal_nnrc,s7spmx)
               chw = cuhw
               qconvey_hlsb = -1.0
               IF (chw .GT. ctw) THEN
                  call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach
     $            ,ifacteaa)
               ELSE
                  qconvey_hlsb = 0.0
               ENDIF
               qconvey_total = qconvey_nnrc + qconvey_hlsb
            ENDIF
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               rmax_cap_ws = qconvey_total - add_conv_red_nnrhl
               dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd
     $         ,rmax_cap_ws)
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del
     $         ),0.0) * frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
```

```
     $         supply_to_nnrc,0.0)
               add_conv_red_hill = add_conv_red_nnrhl - dmnd_nnr_orig +
     $         dmnd_nnr_aft_resasr
               rem_pot_supply_excess_hillcap=amax1
     $         (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
               dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            ELSE
               dmnd_nnr = 0.0
            ENDIF
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            nnrc_cap_upper_grav = amin1(qconvey_nnrc ,total_nnrc_supply)
c
c  hillsboro canal
c
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,0.0)
     $         /86400.
            ELSE
               dmnd_hill = 0.0
            ENDIF
            dmnd_hill = amax1(dmnd_hill,0.0)
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   80    qconvey_nnrc = amax1(qconvey_nnrc,0.0)
         qconvey_hlsb = amax1(qconvey_hlsb,0.0)
         qconvey_total = amax1(qconvey_total,0.0)
         qconvey = amin1(s351cap,qconvey_total)
         eaa_convey(3,1) = qconvey_nnrc
         eaa_convey(3,2) = stagelo
         eaa_convey(3,3) = cuhw
         eaa_convey(3,4) = cutw
         eaa_convey(3,5) = clhw
         eaa_convey(3,6) = cltw
         eaa_convey(3,7) = s7_tailw
c
         eaa_convey(4,1) = qconvey_hlsb
         eaa_convey(4,2) = stagelo
         eaa_convey(4,3) = chw
         eaa_convey(4,4) = ctw
c
c
c  when simulating ability to pump water supply to WCAS and LEC
c  check IF headwater at existing s8 spillway is greater THEN
c  intake level of pump. If greater,use the pump;otherwise
c  use the spillway.
c
         IF (eaa_conv_opt_s7 .EQ. 'PUMP') THEN
            IF (eaa_convey(3,6) .LE. pmp_int(is7bsn-6)) THEN
               DO i = 1,7
                  eaa_convey(5,i) = eaa_convey(3,i)
               END DO
               DO i = 1,4
                  eaa_convey(6,i) = eaa_convey(4,i)
               END DO
c
               nnrc_cap_upper_pump = nnrc_cap_upper_grav
```

```
               nnrc_cap_lower_pump = nnrc_cap_lower_grav
               igravity_s7 = 1
               GO TO 120
            ELSE
               igravity_s7 = 0
            ENDIF
         ENDIF
c
c>>>>>>>> PROFILES 5 + 6 (proc4p)
c         READ(10,'(2i9,3f9.0)',end=123)
c     $      iday,month,stagelo,chdep(1),s8pmphw,chdep(2)
c note: chdep(1), s8pmphw + chdep(2) are not needed in this run
c for north new river (pumped flow) and hillsboro (pumped flow) canals
         IF (eaa_conv_opt_s7 .EQ. 'PUMP') THEN
            iagbsn = 2
            dschg_to_meet_s351_dmnd = 0.0
            DO IW = 1,n_days_week_del_eaa(indx_lok_del)
               IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $         ,indx_lok_del)) THEN
                  dschg_to_meet_s351_dmnd= sum_eaa_dmnd_ssm_nc(is7bsn
     $            -6)* frac_eaa_dmnd_met_nc
               ENDIF
            ENDDO
c           dschg_to_meet_s351_dmnd= amax1(agp(is7bsn-6),0.0)*pctmet
c    +      /86400.
            IF (.not. LFORPUMP(is7bsn-6)) THEN
            IF (stagelo.LE.pmp_int(innr)) THEN
               qconvey_nnrc = 0.0
               qconvey_hlsb = 0.0
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,cnlcap_ws
     $         (innr) + cnlcap_ws(ihill))
c
               IF (dschg_to_eaa .GT. 0.0) THEN
                  cuhw = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
                  cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
                  cuhw = amin1(cuhw,rmaxstg(is7bsn-6))
                  IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
                     dschg_to_eaa = -1.0
                     call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
                  ENDIF
               ELSE
                  cuhw = stagelo
               ENDIF
               cutw = -1.0
               IF (dschg_to_meet_s351_dmnd .GT. 0.0) THEN
                  ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               ELSE
                  ratio_ws = 0.0
               ENDIF
               supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del
     $         ),0.0) * frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $         supply_to_nnrc,0.0)
               rem_pot_supply_excess_hillcap =amax1
     $         (rem_pot_supply_to_nnrhl - cnlcap_ws(ihill),0.0)
```

```
                dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
                total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $          dmnd_nnr_aft_resasr
                IF (nint(total_nnrc_supply) .GT. 0) THEN
                   call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $             ifacteaa)
                ELSE
                   cutw = cuhw
                ENDIF
                nnrc_cap_upper_pump = total_nnrc_supply
                nnrc_cap_lower_pump = 0.0
                clhw = cutw
                cltw = cutw
                s351_tw = cuhw
                s351cap = dschg_to_eaa
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c  hillsboro canal
c
                IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
                   dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,
     $             0.0)/86400.
                ELSE
                   dmnd_hill = 0.0
                ENDIF
                dmnd_hill = amax1(dmnd_hill,0.0)
                supp_to_hill_frm_res = add_conv_red_nnrhl -
     $          dmnd_nnr_orig + dmnd_nnr_aft_resasr
                supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
                total_supp_to_hill = supp_to_hill_frm_res + dmnd_hill
                IF (nint(total_supp_to_hill) .GT. 0) THEN
                   ctw = -1.0
                   call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $             ifacteaa)
                ELSE
                   ctw = cuhw
                ENDIF
                chw = cuhw
                s6_hw = ctw
                goto 110
             ENDIF
             IF (stagelo .GT. rmaxstg(innr)) THEN
                cuhw = rmaxstg(innr)
             ELSE
                cuhw = stagelo
                s351cap = 0.0
             ENDIF
             chw = cuhw
             sltw = s7_tailw
             IF (cuhw .GT. pmp_int(innr)) THEN
                cltw = pmp_int(innr)
             ELSE
                qconvey_nnrc = 0.0
                qconvey_hlsb = 0.0
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,cnlcap_ws
```

```
     $         (innr) + cnlcap_ws(ihill))
               IF (dschg_to_eaa .GT. 0.0) THEN
                  cuhw = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
                  cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
                  cuhw = amin1(cuhw,rmaxstg(is7bsn-6))
                  IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
                     dschg_to_eaa = -1.0
                     call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
                  ENDIF
               ELSE
                  cuhw = stagelo
               ENDIF
               cutw = -1.0
               IF (dschg_to_meet_s351_dmnd .GT. 0.0) THEN
                  ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               ELSE
                  ratio_ws = 0.0
               ENDIF
               supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel(indx_lok_del
     $         ),0.0)* frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $         supply_to_nnrc,0.0)
               rem_pot_supply_excess_hillcap =amax1
     $         (rem_pot_supply_to_nnrhl - cnlcap_ws(ihill),0.0)
               dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
               total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $         dmnd_nnr_aft_resasr
               IF (nint(total_nnrc_supply) .GT. 0) THEN
                  call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $            ifacteaa)
               ELSE
                  cutw = cuhw
               ENDIF
               nnrc_cap_upper_pump = total_nnrc_supply
               nnrc_cap_lower_pump = 0.0
               clhw = cutw
               cltw = cutw
               s351_tw = cuhw
               s351cap = dschg_to_eaa
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               clhw = cutw
               cltw = cutw
c
c  hillsboro canal
c
               IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
                  dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,
     $            0.0)/86400.
               ELSE
                  dmnd_hill = 0.0
               ENDIF
               dmnd_hill = amax1(dmnd_hill,0.0)
               supp_to_hill_frm_res = add_conv_red_nnrhl -
     $         dmnd_nnr_orig + dmnd_nnr_aft_resasr
               supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
               total_supp_to_hill = supp_to_hill_frm_res + dmnd_hill
```

```
               IF (nint(total_supp_to_hill) .GT. 0) THEN
                  ctw = -1.0
                  call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $            ifacteaa)
               ELSE
                  ctw = cuhw
               ENDIF
               chw = cuhw
               s6_hw = ctw
               goto 110
            ENDIF
            ctw = pmp_int(ihill)
            istype = 1
            istrm = 4
            istrl = 2
            istr_eaa = 6
            ireachu = 6
            ireachl = 9
            ireach = 7
            delta0 = (cuhw-cltw)/25.
            delta1 = delta0
c
            s351cap = -1.0
            call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
c routine eaa_prof_two_reaches(q,cuhw,cltw,cutw,clhw,ireachu,
c                              istrm,ireachl,iagbsn,itermax,ifacteaa)
            qconvey_nnrc = -1.0
            call eaa_prof_two_reaches(qconvey_nnrc,cuhw,cltw,cutw,clhw,
     $      ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
c use the same nnr canal headwater for hlsb
            chw = cuhw
            qconvey_hlsb = -1.0
            IF (chw .GT. ctw) THEN
               call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,
     $         ifacteaa)
            ELSE
               qconvey_hlsb = 0.0
            ENDIF
            qconvey_nnrc = amax1(qconvey_nnrc,0.0)
            qconvey_hlsb = amax1(qconvey_hlsb,0.0)
            qconvey_total = qconvey_nnrc + qconvey_hlsb
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (s351cap .GE. qconvey_total) goto 100
c
c  decrease cuhw (=s351tw), thus increasing s351cap and decreasing
c  both qconvey_nnrc and qconvey_hlsb
            iter = 0
   90       cuhw = cuhw - delta1
            s351cap = -1.0
            call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
            qconvey_nnrc = -1.0
            call eaa_prof_two_reaches(qconvey_nnrc,cuhw,cltw,cutw,clhw,
     $      ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
c use the same nnr canal headwater for hlsb
            chw = cuhw
```

```
            qconvey_hlsb = -1.0
            IF (chw .GT. ctw) THEN
               call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,
     $         ifacteaa)
            ELSE
               qconvey_hlsb = 0.0
            ENDIF
            qconvey_total = qconvey_nnrc + qconvey_hlsb
c
            iter = iter + 1
            IF (iter .EQ. itermax)goto 100
c
            IF (qconvey_total-s351cap .GT. tolerq/5.) THEN
c CONTINUE to decrease cuhw such that s351cap increases and qconvey_total decrea
               goto 90
            ENDIF
            IF (s351cap-qconvey_total .GT. tolerq/5.) THEN
c overshot estimation of new cuhw; redo with 1/2 delta1
               cuhw = cuhw + delta1
               delta1 = delta1/2.
               goto 90
            ENDIF
c adjustment within tolerance; exit
  100       dschg_to_meet_s351_dmnd = 0.0
            DO IW = 1,n_days_week_del_eaa(indx_lok_del)
               IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $         ,indx_lok_del)) THEN
                  dschg_to_meet_s351_dmnd= sum_eaa_dmnd_ssm_nc(is7bsn
     $            -6) * frac_eaa_dmnd_met_nc
               ENDIF
            ENDDO
c 100       dschg_to_meet_s351_dmnd= amax1(agp(is7bsn-6),0.0)*pctmet
c    +      /86400.
            conv_max_ws = amax1(cnlcap_ws(innr) + cnlcap_ws(ihill) ,
     $      qconvey_total)
            dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd ,conv_max_ws)

            IF (dschg_to_eaa .GT. s351cap .OR. dschg_to_meet_s351_dmnd
     $      .GT. amax1(qconvey_total-add_conv_red_nnrhl,0.0)) THEN
               cuhw = -1.0
               call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               cuhw = amax1(cuhw,rmax_tw_eaad(is7bsn-6))
               IF ( cuhw .EQ. rmax_tw_eaad(is7bsn-6)) THEN
                  dschg_to_eaa = -1.0
                  call eaaspill(dschg_to_eaa,stagelo,cuhw,6,ifacteaa)
               ENDIF
               s351cap = dschg_to_eaa
               cutw = -1.0
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               supply_to_nnrc = ratio_ws * amax1(alocnnr,0.0)
     $         * frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = dschg_to_eaa - supply_to_nnrc

               add_conv_red_hill = add_conv_red_nnrhl - dmnd_nnr_orig +
     $         dmnd_nnr_aft_resasr
               rmax_cap_ws = amax1(cnlcap_ws(ihill),qconvey_hlsb -
     $         add_conv_red_hill)
```

```
               rem_pot_supply_excess_hillcap =amax1
     $         (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
               dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
               total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $         dmnd_nnr_aft_resasr
               call eaa_prof_lu_table(total_nnrc_supply,cuhw ,cutw,6,
     $         ifacteaa)
               nnrc_cap_upper_pump = total_nnrc_supply
               chw = cutw
               ctw = pmp_int(is7bsn-6)
               delta0 = (chw-ctw)/25.
               call us_flow(1,4,9,iagbsn,chw,ctw,cutw,delta0 ,1.0,100,
     $         nnrc_cap_lower_pump,s7_cap_new ,clhw,ifacteaa)

c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c  hillsboro canal
c
               IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
                  dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,
     $            0.0)/86400.
               ELSE
                  dmnd_hill = 0.0
               ENDIF
               dmnd_hill = amax1(dmnd_hill,0.0)
               supp_to_hill_frm_res = add_conv_red_nnrhl -
     $         dmnd_nnr_orig + dmnd_nnr_aft_resasr
               supp_to_hill_frm_res = amax1(supp_to_hill_frm_res,0.0)
               dischg_to_hill = dmnd_hill
               total_supp_to_hill = supp_to_hill_frm_res +
     $         dischg_to_hill
               ctw = -1.0
               call eaa_prof_lu_table(total_supp_to_hill,cuhw,ctw, 7,
     $         ifacteaa)
               s6_hw = ctw
               chw = cuhw
            ELSE
               IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
                  ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
                  supply_to_nnrc=ratio_ws*amax1(alocnnr_potdel
     $            (indx_lok_del),0.0)*frac_eaa_dmnd_met_nc
                  rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $            supply_to_nnrc,0.0)
                  add_conv_red_hill = add_conv_red_nnrhl -
     $            dmnd_nnr_orig + dmnd_nnr_aft_resasr
                  rmax_cap_ws = amax1(cnlcap_ws(ihill),qconvey_hlsb -
     $            add_conv_red_hill)
                  rem_pot_supply_excess_hillcap=amax1
     $            (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
                  dmnd_nnr = supply_to_nnrc +
     $            rem_pot_supply_excess_hillcap
               ELSE
                  dmnd_nnr = 0.0
               ENDIF
               total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $         dmnd_nnr_aft_resasr
               nnrc_cap_upper_pump = amax1(qconvey_nnrc ,
     $         total_nnrc_supply)
```

```
            ENDIF
        ELSE
c  forward pumping when LOK is low
            ctw = pmp_int(ihill)
            istype = 1
            istrm = 4
            istrl = 2
            istr_eaa = 10
            ireachu = 6
            ireachl = 9
            ireach = 7
            rmax_cuhw = amin1(rmaxstg(is7bsn-6)
     $      ,stagelo+rmax_lift(is7bsn-6))
            cuhw = rmax_cuhw
            delta0 = (cuhw-cltw)/25.
            delta1 = delta0
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c  decrease cuhw (=s351tw), thus increasing s351cap and decreasing
c  both qconvey_nnrc and qconvey_hlsb
            iter = 0
  105       cuhw = cuhw - delta1
            s351cap = -1.0
            call eaaspill(s351cap,stagelo,cuhw,istr_eaa,ifacteaa)
c
            qconvey_nnrc = -1.0
c
c routine eaa_prof_two_reaches(q,cuhw,cltw,cutw,clhw,ireachu,
c
            call eaa_prof_two_reaches(qconvey_nnrc,cuhw,cltw,cutw,clhw,
     $      ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
c use the same nnr canal headwater for hlsb
            chw = cuhw
            qconvey_hlsb = -1.0
            IF (chw .GT. ctw) THEN
               call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,
     $         ifacteaa)
            ELSE
               qconvey_hlsb = 0.0
            ENDIF
            qconvey_total = qconvey_nnrc + qconvey_hlsb
c
            iter = iter + 1
            IF (iter .EQ. itermax)goto 108
c
            IF (qconvey_total-s351cap .GT. tolerq/5.) THEN
c CONTINUE to decrease cuhw such that s351cap increases and qconvey_total decrea
               goto 105
            ENDIF
            IF (s351cap-qconvey_total .GT. tolerq/5.) THEN
c overshot estimation of new cuhw; redo with 1/2 delta1
               cuhw = cuhw + delta1
               delta1 = delta1/2.
               goto 105
            ENDIF
            IF (cuhw .gt. rmax_cuhw) then
```

```
c
               cuhw = rmax_cuhw
               qconvey_nnrc = -1.0
               call eaa_prof_two_reaches(qconvey_nnrc,cuhw,cltw,cutw,
     $         clhw,ireachu,istrm,ireachl,iagbsn,itermax,ifacteaa)
c use the same nnr canal headwater for hlsb
               chw = cuhw
               qconvey_hlsb = -1.0
               IF (chw .GT. ctw) THEN
                  call eaa_prof_lu_table(qconvey_hlsb,chw,ctw,ireach,
     $            ifacteaa)
               ELSE
                  qconvey_hlsb = 0.0
               ENDIF
               qconvey_total = qconvey_nnrc + qconvey_hlsb
            ENDIF
  108       IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               rmax_cap_ws = qconvey_total - add_conv_red_nnrhl
               dschg_to_eaa = amin1(dschg_to_meet_s351_dmnd
     $         ,rmax_cap_ws)
               ratio_ws = dschg_to_eaa/dschg_to_meet_s351_dmnd
               supply_to_nnrc = ratio_ws *amax1(alocnnr_potdel
     $         (indx_lok_del),0.0)* frac_eaa_dmnd_met_nc
               rem_pot_supply_to_nnrhl = amax1(dschg_to_eaa -
     $         supply_to_nnrc,0.0)
               add_conv_red_hill = add_conv_red_nnrhl - dmnd_nnr_orig +
     $         dmnd_nnr_aft_resasr
               rem_pot_supply_excess_hillcap=amax1
     $         (rem_pot_supply_to_nnrhl - rmax_cap_ws,0.0)
               dmnd_nnr = supply_to_nnrc + rem_pot_supply_excess_hillcap
            ELSE
               dmnd_nnr = 0.0
            ENDIF
            total_nnrc_supply = dmnd_nnr + dmnd_nnr_orig -
     $      dmnd_nnr_aft_resasr
            nnrc_cap_upper_pump = amin1(qconvey_nnrc ,total_nnrc_supply)
c
c  hillsboro canal
c
            IF (nint(dschg_to_meet_s351_dmnd) .GT. 0) THEN
               dmnd_hill = dschg_to_eaa - dmnd_nnr - amin1(alocnnr,0.0)
     $         /86400.
            ELSE
               dmnd_hill = 0.0
            ENDIF
            dmnd_hill = amax1(dmnd_hill,0.0)
         ENDIF
  110       qconvey_nnrc = amax1(qconvey_nnrc,0.0)
            qconvey_hlsb = amax1(qconvey_hlsb,0.0)
            qconvey_total = amax1(qconvey_total,0.0)
            qconvey = amin1(s351cap,qconvey_total)
            eaa_convey(5,1) = qconvey_nnrc
            eaa_convey(5,2) = stagelo
            eaa_convey(5,3) = cuhw
            eaa_convey(5,4) = cutw
            eaa_convey(5,5) = clhw
            eaa_convey(5,6) = cltw
```

```
            eaa_convey(6,1) = qconvey_hlsb
            eaa_convey(6,2) = stagelo
            eaa_convey(6,3) = chw
            eaa_convey(6,4) = ctw
         ENDIF
      ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c>>>>>>>> PROFILE 7 (proc3)
c         READ(10,'(2i9,3f9.0)',end=123)
c     $      iday,month,stagelo,chdep(1),s8pmphw,chdep(2)
c note: s8pmphw is used as structure headwater for s-7 in this context
c
c for west palm beach canal
c
  120 iagbsn = 3
      IF (itw_opt(is5a_no_sim) .EQ. 0) THEN
         s5a_tailw = chdep(idn_canal_index(is5a_no_sim))
      ELSE
         dpthgw = (ells(node_tw(is5a_no_sim)) - h(node_tw(is5a_no_sim)))
     $   *s(node_tw(is5a_no_sim))
         IF (pond(node_tw(is5a_no_sim)) .GT. dpthgw) THEN
            s5a_tailw = pond(node_tw(is5a_no_sim)) - dpthgw + ells
     $      (node_tw(is5a_no_sim))
         ELSE
            s5a_tailw = (pond(node_tw(is5a_no_sim)) - dpthgw)/s(node_tw
     $      (is5a_no_sim)) + ells(node_tw(is5a_no_sim))

         ENDIF
      ENDIF
      add_conv_red_wpb = 0.0
      DO irs = 1,nres_ws
         IF (iconv_use_opt_ws(irs) .EQ. 1) THEN
            DO js = 1,noutlets(irs)
               IF (ieaa_bsn(irs,js) .EQ. is5absn-6) THEN
                  add_conv_red_wpb = add_conv_red_wpb +
     $            res_outflow_to_basin(irs,js)
               ENDIF
            END DO
         ENDIF
      END DO
      dschg_to_meet_s352_dmnd = 0.0
      DO IW = 1,n_days_week_del_eaa(indx_lok_del)
         IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $   ,indx_lok_del)) THEN
            dschg_to_meet_s352_dmnd= sum_eaa_dmnd_ssm_nc(is5absn-6)
     $      * frac_eaa_dmnd_met_nc
         ENDIF
      ENDDO
c     dschg_to_meet_s352_dmnd = amax1(agp(is5absn-6),0.0) *pctmet/86400.

      IF (.not. LFORPUMP(is5absn-6)) THEN
      IF (stagelo .GT. rmaxstg(iwpb)) THEN
         chw = rmaxstg(iwpb)
         s352cap = -1.0
         call eaaspill(s352cap,stagelo,chw,7,ifacteaa)
      ELSE
         chw = stagelo
```

```
         s352cap = 0.0
      ENDIF
      stw = s5a_tailw
      IF (chw .GT. pmp_int(iwpb)) THEN
         ctw = pmp_int(iwpb)
      ELSE
         qconvey = 0.0
         qcanal = 0.0
         s5apmpmx = 0.0
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
         dschg_to_eaa = amin1(dschg_to_meet_s352_dmnd,cnlcap(iwpb))
c
         IF (dschg_to_eaa .GT. 0.0) THEN
            chw = -1.0
            call eaaspill(dschg_to_eaa,stagelo,chw,7,ifacteaa)
            chw = amin1(chw,rmaxstg(iwpb))
            total_wpb_supp = dschg_to_eaa + add_conv_red_wpb
         ELSE
            total_wpb_supp = add_conv_red_wpb
            chw = stagelo
         ENDIF
         ctw = -1.0
         call eaa_prof_lu_table(total_wpb_supp,chw ,ctw,4,ifacteaa)

         s352_tw = chw
         chwfinal = chw
         s352cap = dschg_to_eaa
         s5a_hw = ctw
c calculate actual gate opening
         IF (stagelo .GT. chwfinal) THEN
            gateop1 = dschg_to_eaa /(276.0*sqrt(stagelo-chwfinal))
            IF (ifacteaa .EQ. 1) THEN
               gateop1 = gateop1 / rcpfacteaa(is5absn-6)
            ENDIF
         ELSE
            gateop1 = 0.0
         ENDIF
c
         IF (gateop1 .LE. 6.3) THEN
            go_s352 = gateop1
         ELSE
            go_s352 = 6.3
         ENDIF
c
         goto 130
      ENDIF
      istype = 1
      istr_eaa = 7
      ireach = 4
      delta0 = (chw-ctw)/25.
      chwmax = chw
c
c routine us_flow(istype,istr_eaa,ireach,iagbsn,
c  +  chw,ctw,shw,delta0,toler,itermax,canalq,strq,stw,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      call us_flow(istype,istr_eaa,ireach,iagbsn, chw,ctw,stagelo,
```

```
     $delta0,tolerq, itermax,canalq,s352q,stw,ifacteaa)

      IF (chw .GE. chwmax) THEN
c s352 tw limit is exceeded by optimum stage
         chwfinal = chwmax
         s352cap = -1.0
         call eaaspill(s352cap,stagelo,chwfinal,7,ifacteaa)
         qcanal = -1.0
         IF (chwfinal .GT. ctw) THEN
            call eaa_prof_lu_table(qcanal,chwfinal,ctw, ireach,ifacteaa)

         ELSE
            qcanal = 0.0
         ENDIF
      ELSE
         chwfinal = chw
         s352cap = s352q
         qcanal = canalq
      ENDIF
      s5apmpmx = 99999.99
      qcanal = amax1(qcanal,0.0)
c
      qconvey = amin1(s352cap,qcanal,s5apmpmx)
c calculate actual gate opening
      IF (stagelo .GT. chwfinal) THEN
         gateop1 = qconvey /(276.0*sqrt(stagelo-chwfinal))
         IF (ifacteaa .EQ. 1) THEN
            gateop1 = gateop1 / rcpfacteaa(is5absn-6)
         ENDIF
      ELSE
         gateop1 = 0.0
      ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      conv_max_ws = amax1(cnlcap(iwpb),qcanal-add_conv_red_wpb)
      dschg_to_eaa = amin1(dschg_to_meet_s352_dmnd,conv_max_ws)
      IF (dschg_to_eaa .GT. s352cap .OR. dschg_to_meet_s352_dmnd .GT.
     $amax1(qcanal-add_conv_red_wpb ,0.0)) THEN

         chw = -1.0
         call eaaspill(dschg_to_eaa,stagelo,chw,7,ifacteaa)
         chw = amax1(chw,rmax_tw_eaad(is5absn-6))
         IF ( chw .EQ. rmax_tw_eaad(is5absn-6)) THEN
            dschg_to_eaa = -1.0
            call eaaspill(dschg_to_eaa,stagelo,chw,7,ifacteaa)
         ENDIF
         chwfinal = chw
         total_wpb_supp = dschg_to_eaa + add_conv_red_wpb
         ctw = -1.0
         call eaa_prof_lu_table(total_wpb_supp,chw ,ctw,4,ifacteaa)

         s352cap = dschg_to_eaa
         s5a_hw = ctw
c calculate actual gate opening
         IF (stagelo .GT. chwfinal) THEN
            gateop1 = dschg_to_eaa/(276.0*sqrt(stagelo-chwfinal))
            IF (ifacteaa .EQ. 1) THEN
               gateop1 = gateop1 / rcpfacteaa(is5absn-6)
```

```
            ENDIF
         ELSE
            gateop1 = 0.0
         ENDIF
      ENDIF
c
      IF (gateop1.LE. 6.3) THEN
         go_s352 = gateop1
      ELSE
         go_s352 = 6.3
      ENDIF
  130 CONTINUE
      ELSE
         istype = 1
         istr_eaa = 7
         ireach = 4
         s5apmpmx = 999999.
         cuhw = amin1(stagelo+rmax_lift(is5absn-6)
     $   ,rmaxstg(is5absn-6))
         s352cap = -1.0
         call eaaspill (s352cap,stagelo,cuhw,11,ifacteaa)
         chwmax = amin1(stagelo+rmax_lift(is5absn-6)
     $   ,rmaxstg(is5absn-6))
         qcanal = forw_pump_cap(is5absn-6)
         ctw = pmp_int(is5absn-6)
         chwfinal = -1.0
         call eaa_prof_lu_table(qcanal,chwfinal,ctw,ireach,ifacteaa)
         IF (chwfinal .gt. chwmax) THEN
           chwfinal = chwmax
           qcanal = -1.0
           call eaa_prof_lu_table(qcanal,chwfinal,ctw,ireach,ifacteaa)
         ENDIF
         qconvey = amin1(s352cap,qcanal,s5apmpmx)
      ENDIF
      eaa_convey(7,1) = qconvey
      eaa_convey(7,2) = stagelo
      eaa_convey(7,3) = chwfinal
      eaa_convey(7,4) = ctw
c<<<<<<<<
      DO IW = 1,n_days_week_del_eaa(indx_lok_del)
         IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx(iw
     $   ,indx_lok_del)) THEN
            sum_eaa_dmnd_ssm_nc(is8bsn-6) = 0.0
            sum_eaa_dmnd_ssm_nc(is7bsn-6) = 0.0
            sum_eaa_dmnd_ssm_nc(is5absn-6) = 0.0
         ENDIF
      ENDDO
clmb
  140 RETURN
      END
      REAL function eaapumpq(hw,tw,str_no,iagbsn)
      INTEGER str_no
c
c calculates flow through a pump in the eaa
c  eaa pump name  eaa pump no.
c      s8pmp           1
c      s7pmp           2
```

```
c      s6pmp           3
c      s5apmp          4
c assumption: pumps are able to pump at capacity whatever the
c             hw and tw conditions are, i.e. pumps can handle
c             whatever canal conveyance capacity dictates
c currently, given values are pump design discharges
      goto(10,20,30,40) str_no
   10 eaapumpq = 4170.
      RETURN
   20 eaapumpq = 2490.
      RETURN
   30 eaapumpq = 2925.
      RETURN
   40 eaapumpq = 4600.
      RETURN
      END
      SUBROUTINE eaaspill(qstr,hw,tw,str_no,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c calculates flow through a spillway in the eaa
c  eaa spillway name  eaa spillway no.
c         s8                 1
c         s7                 2
c         s8new              3
c         s7new              4
c         s354               5
c         s351               6
c         s352               7
c         s150               8
c-----------------------------
c
      REAL hw,tw,mintw_gotbl(10),max_go(10)
      INTEGER str_no
c-----------------------------
c COMMON block for actual gate openings
      include 'wmm_par.inc'
      include 'fin_go.inc'
c-----------------------------
      INCLUDE 'lakwca.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'abc.inc'
c
c
      mintw_gotbl(1) = -1.0
      mintw_gotbl(2) = -1.0
      mintw_gotbl(3) = -1.0
      mintw_gotbl(4) = -1.0
      mintw_gotbl(5) = 6.0
      mintw_gotbl(6) = 5.0
      mintw_gotbl(7) = 5.0
c
      max_go(1) = -1.0
      max_go(2) = -1.0
      max_go(3) = -1.0
      max_go(4) = -1.0
      max_go(5) = 8.3
      max_go(6) = 8.0
```

```
      max_go(7) = 6.3
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      goto (10,20,30,40,50,50,50,90,100,110,120) str_no
c southern eaa structures
c spillway st s-8
   10 IF (qstr .EQ. -1.0) THEN
         IF (hw .GT. tw) THEN
            qstr = 1252.*(hw-tw)**0.4225
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is8bsn-6)
      ELSE IF(hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = (qstr1/1252.)**(1./0.4225) + tw
         ELSE
            hw = tw
         ENDIF
      ELSE
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/1252.)**(1./0.4225)
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s8 = -1.0
      RETURN
c spillway st s-7
   20 IF (qstr .EQ. -1.0) THEN
         IF (hw .GT. tw) THEN
            qstr = 834.6*(hw-tw)**0.401
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is7bsn-6)
      ELSE IF(hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = (qstr1/834.6)**(1./0.401) + tw
         ELSE
            hw = tw
         ENDIF
      ELSE
```

```
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/834.6)**(1./0.401)
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s7 = -1.0
      RETURN
c spillway st s-8new
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   30 IF (qstr .EQ. -1.0) THEN
         IF (hw .GT. tw) THEN
            qstr = 2500.*(hw-tw)**0.4225
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is8bsn-6)
      ELSE IF(hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = (qstr1/2500.)**(1./0.4225) + tw
         ELSE
            hw = tw
         ENDIF
      ELSE
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/2500.)**(1./0.4225)
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s8new = -1.0
      RETURN
c spillway st s-7new
   40 IF (qstr .EQ. -1.0) THEN
         IF (hw .GT. tw) THEN
            qstr = 1634.6*(hw-tw)**0.401
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is7bsn-6)
      ELSE IF(hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
```

```
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = (qstr1/1634.6)**(1./0.401) + tw
         ELSE
            hw = tw
         ENDIF
      ELSE
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/1634.6)**(1./0.401)
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s7new = -1.0
      RETURN
c northern eaa structures
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   50 gateop = -1.0
      IF (tw .GT. mintw_gotbl(str_no)) THEN
         call go_lu_table(gateop,hw,tw,str_no)
      ELSE
         gateop = max_go(str_no)
      ENDIF
      gateop = amin1(gateop,max_go(str_no))
      gateop = amax1(gateop,0.0)
      goto (60,70,80) str_no-4
   60 IF (qstr .EQ. -1.0) THEN
         tw_stg = amax1(tw,rmax_tw_eaad(is8bsn-6))
         IF (hw .GT. tw_stg) THEN
            IF (gateop .LT.8.3) THEN
               qstr = 276.*gateop*sqrt(hw-tw_stg)
            ELSE
               qstr = 276.*(tw-3.2)*sqrt(hw-tw_stg)
            ENDIF
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is8bsn-6)
      ELSE IF (hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = tw + (qstr1/(276.0*gateop)) ** 2.0
         ELSE
            hw = tw
         ENDIF
```

```
      ELSE
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is8bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/(276.0*gateop)) ** 2.0
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s354 = gateop
      RETURN
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   70 IF (qstr .EQ. -1.0) THEN
         tw_stg = amax1(tw,rmax_tw_eaad(is7bsn-6))
         IF (hw .GT. tw_stg) THEN
            IF (gateop .LT.8.0) THEN
               qstr = 360.*gateop*sqrt(hw-tw_stg)
            ELSE
               qstr = 360.*(tw-4.5)*sqrt(hw-tw_stg)
            ENDIF
         ELSE
            qstr = 0.0
         ENDIF
         IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is7bsn-6)
      ELSE IF (hw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            hw = tw + (qstr1/(360.0*gateop)) ** 2.0
         ELSE
            hw = tw
         ENDIF
      ELSE
         IF (ifacteaa .EQ. 1) THEN
            qstr1 = qstr/rcpfacteaa(is7bsn-6)
         ELSE
            qstr1 = qstr
         ENDIF
         IF (qstr1 .GT. 0.0) THEN
            tw = hw - (qstr1/(360.0*gateop)) ** 2.0
         ELSE
            tw = hw
         ENDIF
      ENDIF
      go_s351 = gateop
      RETURN
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   80 IF (qstr .EQ. -1.0) THEN
         tw_stg = amax1(tw,rmax_tw_eaad(is5absn-6))
         IF (hw .GT. tw_stg) THEN
            IF (gateop .LT. 6.3) THEN
```

```
            qstr = 276.*gateop*sqrt(hw-tw_stg)
         ELSE
            qstr = 276.*(tw-5.2)*sqrt(hw-tw_stg)
         ENDIF
      ELSE
         qstr = 0.0
      ENDIF
      IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is5absn-6)
   ELSE IF (hw .EQ. -1.0) THEN
      IF (ifacteaa .EQ. 1) THEN
         qstr1 = qstr/rcpfacteaa(is5absn-6)
      ELSE
         qstr1 = qstr
      ENDIF
      IF (qstr1 .GT. 0.0) THEN
         hw = tw + (qstr1/(276.0*gateop)) ** 2.0
      ELSE
         hw = tw
      ENDIF
   ELSE
      IF (ifacteaa .EQ. 1) THEN
         qstr1 = qstr/rcpfacteaa(is5absn-6)
      ELSE
         qstr1 = qstr
      ENDIF
      IF (qstr1 .GT. 0.0) THEN
         tw = hw - (qstr1/(276.0*gateop)) ** 2.0
      ELSE
         tw = hw
      ENDIF
   ENDIF
   go_s352 = gateop
   RETURN
90 IF (qstr .EQ. -1.0) THEN
      IF (hw .GT. tw) THEN
         qstr = 1000.0*sqrt(hw-tw)
      ELSE
         qstr = 0.0
      ENDIF
      IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is7bsn-6)
   ELSE IF (hw .EQ. -1.0) THEN
      IF (ifacteaa .EQ. 1) THEN
         qstr1 = qstr/rcpfacteaa(is7bsn-6)
      ELSE
         qstr1 = qstr
      ENDIF
      IF (qstr1 .GT. 0.0) THEN
         hw = tw + (qstr1/1000.0) ** 2.0
      ELSE
         hw = tw
      ENDIF
   ELSE
      IF (ifacteaa .EQ. 1) THEN
         qstr1 = qstr/rcpfacteaa(is7bsn-6)
      ELSE
         qstr1 = qstr
      ENDIF
```

```
      IF (qstr1 .GT. 0.0) THEN
         tw = hw - (qstr1/1000.0) ** 2.0
      ELSE
         tw = hw
      ENDIF
   ENDIF
   RETURN
100 IF (qstr .EQ. -1.0) THEN
      qstr = forw_pump_cap(is8bsn-6)
      IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is8bsn-6)
      IF (stagelo .GT. rmin_lok_stg_forw_pump(is8bsn-6)+0.1) THEN
         qstr1 = qstr
      ELSEIF(stagelo .GT. rmin_lok_stg_forw_pump(is8bsn-6)) THEN
         qstr1 = 10.0 * (stagelo-rmin_lok_stg_forw_pump(is8bsn-6)
  $      ) * qstr
      ELSE
         qstr1 = 0.0
      ENDIF
      qstr = qstr1
   ELSE IF (hw .EQ. -1.0) THEN
      hw = stagelo
   ELSE
      tw = amin1(stagelo+rmax_lift(is8bsn-6),rmaxstg(is8bsn-6))
   ENDIF
   RETURN
110 IF (qstr .EQ. -1.0) THEN
      qstr = forw_pump_cap(is7bsn-6)
      IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is7bsn-6)
      IF (stagelo .GT. rmin_lok_stg_forw_pump(is7bsn-6)+0.1) THEN
         qstr1 = qstr
      ELSEIF(stagelo .GT. rmin_lok_stg_forw_pump(is7bsn-6)) THEN
         qstr1 = 10.0 * (stagelo-rmin_lok_stg_forw_pump(is7bsn-6)
  $      ) * qstr
      ELSE
         qstr1 = 0.0
      ENDIF
      qstr = qstr1
   ELSE IF (hw .EQ. -1.0) THEN
      hw = stagelo
   ELSE
      tw = amin1(stagelo+rmax_lift(is7bsn-6),rmaxstg(is7bsn-6))
   ENDIF
   RETURN
120 IF (qstr .EQ. -1.0) THEN
      qstr = forw_pump_cap(is5absn-6)
      IF (ifacteaa .EQ. 1) qstr = qstr*rcpfacteaa(is5absn-6)
      IF (stagelo .GT. rmin_lok_stg_forw_pump(is5absn-6)+0.1) THEN
         qstr1 = qstr
      ELSEIF(stagelo .GT. rmin_lok_stg_forw_pump(is5absn-6)) THEN
         qstr1 = 10.0 * (stagelo-rmin_lok_stg_forw_pump(is5absn-6)
  $      ) * qstr
      ELSE
         qstr1 = 0.0
      ENDIF
      qstr = qstr1
   ELSE IF (hw .EQ. -1.0) THEN
      hw = stagelo
```

```
      ELSE
         tw = amin1(stagelo+rmax_lift(is5absn-6),rmaxstg(is5absn-6))
      ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      RETURN
      END
C
C   This subroutine calculates equilibrium canal headwater and upstream
C   discharge given canal tailwater,dowmstream structure discharge,
C   and headwater for upstream structure in a structure-canal-
C   structure configuration in the EAA.
C
      SUBROUTINE us_flow(istype,istr_eaa,ireach,iagbsn,chw,ctw,shw,
     $delta0,toler,itermax,canalq,strq,stw,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c definition of variables:
c   istype   = structure type
c   istr_eaa = structure number
c   ireach   = reach number
c   iagbsn   = basin number
c   chw      = starting canal headwater
c   ctw      = fixed canal tailwater
c   shw      = fixed upper structure headwater
c   delta0   = initial step size
c   toler    = convergence criterion for matching flows
c   itermax  = maximum iteration count
c   canalq   = equilibrium canal discharge
c   strq     = equilibrium upstream structure discharge
c   stw      = final structure tailwater (or canal headwater)
c
c-----------------------------
      iter = 0
      goto(10,80)istype
cc structure type is spillway
c figure out direction of step
   10 IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (shw .GT. chw) THEN
         strq = -1.0
         call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
      ELSE
         strq = 0.0
      ENDIF
      strq_prev = strq
      IF (strq-canalq)20,30,40
c step is negative; decrease canalq and increase strq
   20 sign = -1.0
      goto 50
c step is zero; canalq and strq are the same
   30 stw = chw
      RETURN
```

```
c step is positive; increase canalq and decrease strq
   40 sign = 1.0
   50 delta = delta0
   60 chw = chw + sign*delta
      IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (shw .GT. chw) THEN
         strq = -1.0
         call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
         IF (sign*(strq - strq_prev) .GT. 0.0.AND.((ireach .GE. 1 .AND.
     $   ireach .LE. 4) .OR. ireach .EQ. 7)) GO TO 70
      ELSE
         strq = 0.0
      ENDIF
      strq_prev = strq
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      iter = iter + 1
      IF (iter .EQ. itermax) goto 70
c
      IF (sign*(strq-canalq) .GT. toler) THEN
c CONTINUE to change chw in the normal trend
         goto 60
      ENDIF
      IF (sign*(canalq-strq) .GT. toler) THEN
c overshot estimation of new chw; redo with 1/2 delta
         chw = chw - sign*delta
         delta = delta/2.
         goto 60
      ENDIF
c adjustment within tolerance; exit
   70 stw = chw
      RETURN
c
cc structure is a pump
c note: pumping is not yet an option here
   80 IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (shw .GT. chw) THEN
         strq = -1.0
         call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
      ELSE
         strq = 0.0
      ENDIF
      IF (strq-canalq)90,100,110
c step is negative; decrease canalq and increase strq
```

```
   90 sign = -1.0
      goto 120
c step is zero; canalq and strq are the same
  100 stw = chw
      RETURN
  110 sign = 1.0
  120 delta = delta0
  130 chw = chw + sign*delta
      IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      strq = -1.0
      call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      iter = iter + 1
      IF (iter .EQ. itermax) goto 140
c
      IF (sign*(strq-canalq) .GT. toler) THEN
c CONTINUE to change chw in the normal trend
         goto 130
      ENDIF
      IF (sign*(canalq-strq) .GT. toler) THEN
c overshot estimation of new chw; redo with 1/2 delta
         chw = chw - sign*delta
         delta = delta/2.
         goto 130
      ENDIF
c adjustment within tolerance; exit
  140 stw = chw
      RETURN
      END
      SUBROUTINE ds_flow(istype,istr_eaa,ireach,iagbsn,chw,ctw,stw,
     $delta0,toler,itermax,canalq,strq,shw,ifacteaa)
c
c definition of variables:
c   istype   = structure type
c   istr_eaa = structure number
c   ireach   = reach number
c   iagbsn   = basin number
c   chw      = fixed canal headwater
c   ctw      = starting canal tailwater
c   stw      = fixed lower structure tailwater
c   delta0   = initial step size
c   toler    = convergence criterion for matching flows
c   itermax  = maximum iteration count
c   canalq   = equilibrium canal stage
c   strq     = equilibrium downstream structure discharge
c   shw      = final structure headwater (or canal tailwater)
c
c-----------------------------
      iter = 0
      goto(10,80)istype
```

```
cc structure is a spillway
c figure out direction of step
   10 IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (ctw .GT. stw) THEN
         strq = -1.0
         call eaaspill(strq,ctw,stw,istr_eaa,ifacteaa)
      ELSE
         strq = 0.0
      ENDIF
      IF (canalq-strq)20,30,40
c step is negative; increase canalq and decrease strq
   20 sign = -1.0
      goto 50
c step is zero; canalq and strq are the same
   30 shw = ctw
      RETURN
c step is positive; decrease canalq and increase strq
   40 sign = 1.0
   50 delta = delta0
   60 ctw = ctw + sign*delta
      IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (ctw .GT. stw) THEN
         strq = -1.0
         call eaaspill(strq,ctw,stw,istr_eaa,ifacteaa)
      ELSE
         strq = 0.0
      ENDIF
c
      iter = iter + 1
      IF (iter .EQ. itermax) goto 70
c
      IF (sign*(canalq-strq) .GT. toler) THEN
c CONTINUE to change ctw in the normal trend
         goto 60
      ENDIF
      IF (sign*(strq-canalq) .GT. toler) THEN
c overshot estimation of new ctw; redo with 1/2 delta
         ctw = ctw - sign*delta
         delta = delta/2.
         goto 60
      ENDIF
c adjustment within tolerance; exit
```

```
   70 shw = ctw
      RETURN
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
cc structure is a pump
c note: pumping is not yet an option here
   80 RETURN
      END
C
C  This subroutine calculates equilibrium interior headwater and
C  tailwater stages and upstream discharges. THis is done for
C  a system with an intermediate structure included; thus have
C  a structure-canal-structure-canal-structure system (e.g.
C  Miami canal system with STA34 included in EAA).
C
      SUBROUTINE usus_flow(istype,istrm,istru,ireachu,ireachl,iagbsn,
     $cuhw,cltw,cutw,clhw,suhw,delta0,toler,itermax,canalq,struq,sutw,
     $ifacteaa)
c
c definition of variables:
c   istype   = structure type
c   istrm    = structure number in the middle of the two reaches
c   istru    = structure number at the upstream end of the upper reach
c   ireachu  = upper reach number
c   ireachl  = lower reach number
c   iagbsn   = basin number
c   cuhw     = starting upper canal headwater
c   cltw     = fixed lower canal tailwater
c   cutw     = final upper canal tailwater
c   clhw     = final lower canal headwater
c   suhw     = fixed upper structure headwater
c   delta0   = initial step size
c   toler    = convergence criterion for matching flows
c   itermax  = maximum iteration count
c   canalq   = equilibrium canal stage
c   struq    = equilibrium upstream structure discharge
c   sutw     = final structure tailwater (or upper canal headwater)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c-----------------------------
      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'

      iter = 0

      if (day1) then
         chw=0.0
         ctw=0.0
         shw=0.0
      endif

      goto(10,80)istype
cc structure type is spillway
c figure out direction of step
   10 IF (cuhw .GT. cltw) THEN
         canalq = -1.0
         call eaa_prof_two_reaches(canalq,cuhw,cltw,cutw,clhw, ireachu,
     $   istrm,ireachl,iagbsn,itermax,ifacteaa)
c following line is heuristic
```

```
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (suhw .GT. cuhw) THEN
         struq = -1.0
         call eaaspill(struq,suhw,cuhw,istru,ifacteaa)
      ELSE
         struq = 0.0
      ENDIF
      strq_prev = struq
      IF (struq-canalq)20,30,40
c step is negative; decrease canalq and increase struq
   20 sign = -1.0
      goto 50
c step is zero; canalq and struq are the same
   30 sutw = cuhw
      RETURN
c step is positive; increase canalq and decrease struq
   40 sign = 1.0
   50 delta = delta0
   60 cuhw = cuhw + sign*delta
      IF (cuhw .GT. cltw) THEN
         canalq = -1.0
         call eaa_prof_two_reaches(canalq,cuhw,cltw,cutw,clhw, ireachu,
     $   istrm,ireachl,iagbsn,itermax,ifacteaa)
c following line is heuristic
         IF (canalq .LE. 0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (suhw .GT. cuhw) THEN
         struq = -1.0
         call eaaspill(struq,suhw,cuhw,istru,ifacteaa)
         IF (sign * (struq - strq_prev) .GT. 0.0) GO TO 70
      ELSE
         struq = 0.0
      ENDIF
      strq_prev = struq
c
      iter = iter + 1
      IF (iter .EQ. itermax) goto 70
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (sign*(struq-canalq) .GT. toler) THEN
c CONTINUE to change cuhw in the normal trend
         goto 60
      ENDIF
      IF (sign*(canalq-struq) .GT. toler) THEN
c overshot estimation of new cuhw; redo with 1/2 delta
         cuhw = cuhw - sign*delta
         delta = delta/2.
         goto 60
      ENDIF
c adjustment within tolerance; exit
   70 sutw = cuhw
      RETURN
c
```

```
cc structure is a pump
c note: pumping is not yet an option here
c step is positive; increase canalq and decrease strq
   80 IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      IF (shw .GT. chw) THEN
         strq = -1.0
         call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
      ELSE
         strq = 0.0
      ENDIF
      IF (strq-canalq)90,100,110
c step is negative; decrease canalq and increase strq
   90 sign = -1.0
      goto 120
c step is zero; canalq and strq are the same
  100 stw = chw
      RETURN
  110 sign = 1.0
  120 delta = delta0
  130 chw = chw + sign*delta
      IF (chw .GT. ctw) THEN
         canalq = -1.0
         call eaa_prof_lu_table(canalq,chw,ctw,ireach,ifacteaa)
c following line is heuristic
         IF (canalq.LE.0.0) canalq = 0.1
      ELSE
         canalq = 0.0
      ENDIF
      strq = -1.0
      call eaaspill(strq,shw,chw,istr_eaa,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      iter = iter + 1
      IF (iter .EQ. itermax) goto 140
c
      IF (sign*(strq-canalq) .GT. toler) THEN
c CONTINUE to change chw in the normal trend
         goto 130
      ENDIF
      IF (sign*(canalq-strq) .GT. toler) THEN
c overshot estimation of new chw; redo with 1/2 delta
         chw = chw - sign*delta
         delta = delta/2.
         goto 130
      ENDIF
c adjustment within tolerance; exit
  140 stw = chw
      RETURN
      END
C
C  This subroutine returns equilibrium interior headwater and
```

```
C  tailwater stages.downstream discharges given upper canal
C  headwater and lower structure canal tailwater for a
C  structure_canal-structure-canal-structure syastem in
C  the EAA ; uses HEC2 lookup tables for canal conveyance
C  capacity.
C
      SUBROUTINE dsds_flow(istype,istrm,istrl,ireachu,ireachl,iagbsn,
     $cuhw,cltw,cutw,clhw,sltw,delta0,toler,itermax,canalq,strlq,slhw,
     $ifacteaa)
c
c definition of variables:
c   istype   = structure type
c   istrm    = structure number in the middle of the two reaches
c   istrl    = structure number at the downstream end of lower reach
c   ireachu  = upper reach number
c   ireachl  = lower reach number
c   iagbsn   = basin number
c   cuhw     = fixed upper canal headwater
c   cltw     = starting lower canal tailwater
c   cutw     = final upper canal tailwater
c   clhw     = final lower canal headwater
c   sltw     = fixed lower structure tailwater
c   delta0   = initial step size
c   toler    = convergence criterion for matching flows
c   itermax  = maximum iteration count
c   canalq   = equilibrium canal stage
c   strlq    = equilibrium downstream structure discharge
c   slhw     = final structure headwater (or lower canal tailwater)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c-----------------------------
      iter = 0
      goto(10,80)istype
cc structure is a spillway
c figure out direction of step
   10 IF (cuhw .GT. cltw) THEN
         canalq = -1.0
         call eaa_prof_two_reaches(canalq,cuhw,cltw,cutw,clhw, ireachu,
     $   istrm,ireachl,iagbsn,itermax,ifacteaa)
      ELSE
         canalq = 0.0
      ENDIF
      IF (cltw .GT. sltw) THEN
         strlq = -1.0
         call eaaspill(strlq,cltw,sltw,istrl,ifacteaa)
      ELSE
         strlq = 0.0
      ENDIF
      IF (canalq-strlq)20,30,40
c step is negative; increase canalq and decrease strlq
   20 sign = -1.0
      goto 50
c step is zero; canalq and strlq are the same
   30 slhw = cltw
      RETURN
c step is positive; decrease canalq and increase strlq
   40 sign = 1.0
   50 delta = delta0
```

```
   60 cltw = cltw + sign*delta
      IF (cuhw .GT. cltw) THEN
         canalq = -1.0
         call eaa_prof_two_reaches(canalq,cuhw,cltw,cutw,clhw, ireachu,
     $   istrm,ireachl,iagbsn,itermax,ifacteaa)
      ELSE
         canalq = 0.0
      ENDIF
      IF (cltw .GT. sltw) THEN
         strlq = -1.0
         call eaaspill(strlq,cltw,sltw,istrl,ifacteaa)
      ELSE
         strlq = 0.0
      ENDIF
c
      iter = iter + 1
      IF (iter .EQ. itermax) goto 70
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (sign*(canalq-strlq) .GT. toler) THEN
c CONTINUE to change cltw in the normal trend
         goto 60
      ENDIF
      IF (sign*(strlq-canalq) .GT. toler) THEN
c overshot estimation of new cltw; redo with 1/2 delta
         cltw = cltw - sign*delta
         delta = delta/2.
         goto 60
      ENDIF
c adjustment within tolerance; exit
   70 slhw = cltw
      RETURN
c
cc structure is a pump
c note: pumping is not yet an option here
   80 RETURN
      END
      SUBROUTINE eaa_prof_two_reaches(q,cuhw,cltw,cutw,clhw,ireachu,
     $istrm,ireachl,iagbsn,itermax,ifacteaa)
c
c This SUBROUTINE RETURNs the flow rate (q); upstream stage (cuhw);
c or downstream stage (cltw); given the values of the other two
c open-channel flow variables for a 2-reach with a spillway structure
c in the middle system.
c
c List of Reaches
c Base Case
c 1. EAA Miami Canal:           S354HW to S8TW
c 2. EAA North New River Canal: S351HW to S7TW
c 3. EAA Hillsboro Canal:       S351TW to S6HW
c 4. EAA West Palm Beach Canal: S352TW to S5AHW
c Future Base
c 5. EAA Miami Canal:           S354TW to S8newHW
c 6. EAA North New River Canal: S351TW to S7newHW
c 7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
c 8. EAA Miami Canal:           S8newTW to S8HW
c 9. EAA North New River Canal: S7newTW to S7HW
c
```

```
c CASE I: q must be -1.0 upon entry to this routine
c  given: downstream stage (cltw), upstream stage (cuhw)
c  find: flow rate (q)
c
c CASE II: cuhw must be -1.0 upon entry to this routine
c  given: downstream stage (cltw), flow rate (q)
c  find: upstream stage (cuhw)
c
c CASE III: cltw must be -1.0 upon entry to this routine
c  given: upstream stage (cuhw), flow rate (q)
c  find: downstream stage (cltw)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c declare variables
c nj = 22
c    = max. number of starting downstream stages in table generated
c      from HEC-2 runs
c nq = 20
c    =  number of flow rates used in table generated from HEC-2 runs
c nreach = 30
c        = max. number of reaches
c note: nj = ntws(ireach)
c       nq = n_eaaflows(ireach)
c dsstage(nj,nreach), usstage(nq,nj,nreach), eaaflow(nq,nj,nreach)
c
c-----------------------------
      INTEGER ireachu,ireachl
      REAL q, cuhw, cltw
      REAL length_u_reach, length_l_reach, l_upper_reach(2),
     $l_lower_reach(2)

c COMMON block for eaa canal water surface profile data
      include 'wmm_par.inc'
      include 'eaacap.inc'
c ff 4 lengths in feet
      l_upper_reach(1) = 101990.
      l_lower_reach(1) = 36400.
      l_upper_reach(2) = 130018.
      l_lower_reach(2) = 21120.
      IF (cltw .EQ. -1.0) THEN
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c PROCEED downstream
c given cuhw and qucanal = q; find cutw
         qucanal = q
         cutw = -1.0
         call eaa_prof_lu_table(qucanal,cuhw,cutw,ireachu,ifacteaa)
c given smhw = cutw and qmstr = qucanal; find smtw
         smhw = cutw
         qmstr = qucanal
         smtw = -1.0
         call eaaspill(qmstr,smhw,smtw,istrm,ifacteaa)
c given clhw = smtw and qlcanal = qmstr; find cltw
         clhw = smtw
         qlcanal = qmstr
         cltw = -1.0
         call eaa_prof_lu_table(qlcanal,clhw,cltw,ireachl,ifacteaa)
      ELSE IF(cuhw .EQ. -1.0) THEN
c
```

```
c PROCEED upstream
c given cltw and qlcanal = q; find clhw
         qlcanal = q
         clhw = -1.0
         call eaa_prof_lu_table(qlcanal,clhw,cltw,ireachl,ifacteaa)
c given smtw = clhw and qmstr = qlcanal; find smhw
         smtw = clhw
         qmstr = qlcanal
         smhw = -1.0
         call eaaspill(qmstr,smhw,smtw,istrm,ifacteaa)
c given cutw = cmhw and qucanal = qmstr; find cuhw
         cutw = smhw
         qucanal = qmstr
         cuhw = -1.0
         call eaa_prof_lu_table(qucanal,cuhw,cutw,ireachu,ifacteaa)
      ELSE
c
c ITERATE between ireachu,istrm and ireachl: given cuhw,cltw; find q
c initialize cutw (=clhw)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         iter = 0
         length_u_reach = l_upper_reach(iagbsn)
         length_l_reach = l_lower_reach(iagbsn)
         cutw = cuhw + (cltw-cuhw) / (length_u_reach+length_l_reach)
     $   *length_u_reach
         clhw = cutw
         smhw = cutw
         smtw = cutw
         IF (cuhw .LE. cltw) THEN
c negative gradient; assume zero flow
            q = 0.0
            qucanal = 0.0
            qmstr = 0.0
            qlcanal = 0.0
c assign bogus numbers for the following variables
            cutw = cuhw
            clhw = cutw
            smhw = cutw
            smtw = cutw
         ELSE
c DO regular iteration going upstream
c figure out direction of step based on initial value of clhw
c find qlcanal
            qlcanal = -1.0
            IF (clhw .GT. cltw) THEN
               call eaa_prof_lu_table(qlcanal,clhw,cltw,ireachl,
     $         ifacteaa)
            ELSE
               qlcanal = 0.0
            ENDIF
c following line is heuristic
            IF (qlcanal .LE. 0.0) qlcanal = 0.1
c find smhw
            smtw = clhw
            qmstr = qlcanal
            smhw = -1.0
            call eaaspill(qmstr,smhw,smtw,istrm,ifacteaa)
```

```
c find cuhwx
            cutw = smhw
            qucanal = qmstr
            cuhwx = -1.0
            call eaa_prof_lu_table(qucanal,cuhwx,cutw,ireachu,ifacteaa)
c compare cuhwx and cuhw
            IF (cuhw-cuhwx)10,20,30
c step is negative: decrease q by decreasing clhw
   10       sign = -1.0
            goto 40
c step is zero; cuhw and cuhwx are the same; exit
   20       RETURN
c step is positive: increase q by increasing clhw
   30       sign = 1.0
   40       delta = (cuhw - cltw)/20.
c toler = convergence criterion for matching stages (cuhwx vs. cuhw)
            toler = 0.005
c find qlcanal
   50       clhw = clhw + sign*delta
            qlcanal = -1.0
            IF (clhw .GT. cltw) THEN
               call eaa_prof_lu_table(qlcanal,clhw,cltw,ireachl,
     $         ifacteaa)
            ELSE
               qlcanal = 0.0
            ENDIF
c following line is heuristic
            IF (qlcanal .LE. 0.0) qlcanal = 0.1
c find smhw
            smtw = clhw
            qmstr = qlcanal
            smhw = -1.0
            call eaaspill(qmstr,smhw,smtw,istrm,ifacteaa)
c find cuhwx
            cutw = smhw
            qucanal = qmstr
            cuhwx = -1.0
            call eaa_prof_lu_table(qucanal,cuhwx,cutw,ireachu,ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            iter = iter + 1
            IF (iter .EQ. itermax) goto 60
c compare cuhwx and cuhw
            IF (sign*(cuhw-cuhwx) .GT. toler) THEN
c CONTINUE to change clhw in the normal trend
               goto 50
            ENDIF
            IF (sign*(cuhwx-cuhw) .GT. toler) THEN
c overshot estimation of new clhw; redo with 1/2 delta
               clhw = clhw - sign*delta
               delta = delta/2.
               goto 50
            ENDIF
c adjustment within tolerance; exit
   60       q = qucanal
            RETURN
         ENDIF
c
```

```
      ENDIF
c
      RETURN
      END
c
      SUBROUTINE eaa_prof_lu_table(q,chw,ctw,ireach,ifacteaa)
c
c This SUBROUTINE RETURNs the flow rate (q); upstream stage (chw);
c or downstream stage (ctw); given the values of the other two
c open-channel flow variables.
c
c List of Reaches
c Base Case
c 1. EAA Miami Canal:           S354HW to S8TW
c 2. EAA North New River Canal: S351HW to S7TW
c 3. EAA Hillsboro Canal:       S351TW to S6HW
c 4. EAA West Palm Beach Canal: S352TW to S5AHW
c Future Base
c 5. EAA Miami Canal:           S354TW to S8newHW
c 6. EAA North New River Canal: S351TW to S7newHW
c 7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
c 8. EAA Miami Canal:           S8newTW to S8HW
c 9. EAA North New River Canal: S7newTW to S7HW
c
c CASE I: q must be -1.0 upon entry to this routine
c  given: downstream stage (ctw), upstream stage (chw)
c         + reach number (ireach)
c  find: flow rate (q)
c
c CASE II: chw must be -1.0 upon entry to this routine
c  given: downstream stage (ctw), flow rate (q)
c         + reach number (ireach)
c  find: upstream stage (chw)
c
c CASE III: ctw must be -1.0 upon entry to this routine
c  given: upstream stage (chw), flow rate (q)
c         + reach number (ireach)
c  find: downstream stage (ctw)
c
c declare variables
c nj = 12
c    = max. number of starting downstream stages in table generated
c      from HEC-2 runs
c nq = 10
c    =  number of flow rates used in table generated from HEC-2 runs
c nreach = 15
c        = max. number of reaches
c note: nj = ntws(ireach)
c       nq = n_eaaflows(ireach)
c dsstage(nj,nreach), usstage(nq,nj,nreach), eaaflow(nq,nj,nreach)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c-----------------------------
      INTEGER ireach
      REAL q, ctw, chw, val(22)
c COMMON block for eaa canal water surface profile data
      include 'wmm_par.inc'
      include 'eaacap.inc'
```

```
      INCLUDE 'lakwca.inc'
c-------------------------------------------------------------
c
      nj = ntws(ireach)
      nq = n_eaaflows(ireach)
c
      IF (ireach .EQ. 5 .OR. ireach .EQ. 1 .OR. ireach .EQ. 8) THEN
         ieaabsn = 1
      ELSE IF (ireach .EQ. 2 .OR. ireach .EQ. 6 .OR. ireach .EQ. 9) THEN
         ieaabsn = 2
      ELSE IF (ireach .EQ. 3 .OR. ireach .EQ. 7) THEN
         ieaabsn = 4
      ELSE
         ieaabsn = 3
      ENDIF
c
      IF (ctw .EQ. -1.0) THEN
c calculate q for each tw level at given hw
         DO i=1,ntws(ireach)
            val(i) = ainterp(chw,usstage(1,i,ireach), eaaflow
     $      (1,i,ireach),nq,0.,0.,0)
         END DO
c
c pick q "lines" where given q falls in between
         IF (ifacteaa .EQ. 1) THEN
            q1 = q / rcpfacteaa_canal(ieaabsn)
         ELSE
            q1 = q
         ENDIF
         call locate(val(1),ntws(ireach),q1,ij)
         IF (ij .EQ. 0 .OR. ij .EQ. ntws(ireach)) THEN
            ijtmp = iabs(ij-1)
            ijtmp1 = ijtmp + 1
         ELSE
            ijtmp = ij
            ijtmp1 = ij + 1
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         xq = val(ijtmp)
         xq1 = val(ijtmp1)
         valx = dsstage(ijtmp,ireach)
         valx1 = dsstage(ijtmp1,ireach)
      ELSE
c pick tw lines where given tw falls in between
         call locate(dsstage(1,ireach), nj, ctw, ij)
         IF (ij .EQ. 0 .OR. ij .EQ. nj) THEN
            ijtmp = iabs(ij-1)
            ijtmp1 = ijtmp + 1
         ELSE
            ijtmp = ij
            ijtmp1 = ij + 1
         ENDIF
c
         IF (q .EQ. -1.0) THEN
c solution for CASE I:
c call q twice for the 2 values of tw lines
            value = ainterp(chw, usstage(1,ijtmp,ireach), eaaflow(1,
```

```
     $      ijtmp,ireach), nq, 0.,0.,0)
            value1 = ainterp(chw, usstage(1,ijtmp1,ireach), eaaflow(1,
     $      ijtmp1,ireach), nq, 0.,0.,0)
         ELSE IF (chw .EQ. -1.0) THEN
            IF (ifacteaa .EQ. 1) THEN
               q1 = q / rcpfacteaa_canal(ieaabsn)
            ELSE
               q1 = q
            ENDIF
c solution for CASE II:
c call hw twice for the 2 values of tw lines
            value = ainterp(q1, eaaflow(1,ijtmp,ireach), usstage(1,
     $      ijtmp,ireach), nq, 0.,0.,0)
            value1 = ainterp(q1, eaaflow(1,ijtmp1,ireach), usstage(1,
     $      ijtmp1,ireach), nq, 0.,0.,0)
         ELSE
            WRITE(*,*) ' Dependent variable should be defined as -1.0'
            STOP
         ENDIF
c
         xval = dsstage(ijtmp,ireach)
         xval1 = dsstage(ijtmp1,ireach)
      ENDIF
c
      IF (q .EQ. -1.0) THEN
c solution for CASE I:
c interpolate between the two q's
         IF (value1 .LT. 0.0) THEN
            q = (value/(chw-xval))*(chw-ctw)
            IF (ctw .GT. chw) q=0.0
         ELSE
            q = value + (ctw-xval)/(xval1-xval)*(value1-value)
         ENDIF
         IF (ifacteaa .EQ. 1) q = q * rcpfacteaa_canal(ieaabsn)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      ELSE IF (chw .EQ. -1.0) THEN
c solution for CASE II:
c interpolate between the two chw's
         IF (ifacteaa .EQ. 1) THEN
            q1 = q / rcpfacteaa_canal(ieaabsn)
         ELSE
            q1 = q
         ENDIF
         IF (value1 .LT. 0.0) THEN
            chw = (value/(q1-xval))*(q1-ctw)
         ELSE
            chw = value + (ctw-xval)/(xval1-xval)*(value1-value)
         ENDIF
      ELSE IF (ctw .EQ. -1.0) THEN
         IF (ifacteaa .EQ. 1) THEN
            q1 = q / rcpfacteaa_canal(ieaabsn)
         ELSE
            q1 = q
         ENDIF
c solution for CASE III:
c interpolate between the two ctw's
         ctw = valx + (q1-xq)/(xq1-xq)*(valx1-valx)
```

```
      ELSE
         WRITE(*,*) ' Dependent variable should be defined as -1.0'
         STOP
      ENDIF
c
      RETURN
      END
      SUBROUTINE eaa_prof_lu_table12(q,hw,tw,ireach,ifacteaa)
c-----------------------------
c note: This SUBROUTINE was written to supplement SUBROUTINE
c       eaa_prof_lu_table where profiles 1 + 2 refer to exterior
c       stages.  Normally, i.e. for reaches 3 through 9, interior
c       stages are required.  Therefore, this SUBROUTINE should
c       only be used for determining the profile for the Miami
c       and North New River canals for the "Base Case" condition.
c-----------------------------
c List of Reaches
c Base Case
c 1. EAA Miami Canal:           S354HW to S8TW
c 2. EAA North New River Canal: S351HW to S7TW
c 3. EAA Hillsboro Canal:       S351TW to S6HW
c 4. EAA West Palm Beach Canal: S352TW to S5AHW
c Future Base
c 5. EAA Miami Canal:           S354TW to S8newHW
c 6. EAA North New River Canal: S351TW to S7newHW
c 7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
c 8. EAA Miami Canal:           S8newTW to S8HW
c 9. EAA North New River Canal: S7newTW to S7HW
c
c CASE I: q must be -1.0 upon entry to this routine
c  given: downstream stage (tw), upstream stage (hw)
c         + reach number (ireach)
c  find: flow rate (q)
c
c CASE II: hw must be -1.0 upon entry to this routine
c  given: downstream stage (tw), flow rate (q)
c         + reach number (ireach)
c  find: upstream stage (hw)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c CASE III: tw must be -1.0 upon entry to this routine
c  given: upstream stage (hw), flow rate (q)
c         + reach number (ireach)
c  find: downstream stage (tw)
c
c Variable Definition
c nreach = 15
c        = max. number of reaches
c nj = ntws(ireach)
c nq = n_eaaflows(ireach)
c dsstage(nj,nreach), usstage(nq,nj,nreach), eaaflow(nq,nj,nreach)
c
      INTEGER ireach
      REAL q, tw, hw
c COMMON block for eaa canal water surface profile data
      include 'wmm_par.inc'
      include 'eaacap.inc'
      INCLUDE 'lakwca.inc'
```

```
c-----------------------------
c
      IF (ireach .EQ. 1) THEN
c Miami canal
         irchup = 5
         irchdn = 8
      ELSE
c North New River canal
         irchup = 6
         irchdn = 9
      ENDIF
      IF (q .EQ. -1.0) THEN
c  solve for discharge
         hwdn = hw
         delta0 = (hw-tw)/25.0
         delta1 = delta0
         tolerq = 0.05
         itermax = 150
         iter = 0
   10    hwdn = hwdn - delta1
         qd = -1.0
         call eaa_prof_lu_table(qd,hwdn,tw,irchdn,ifacteaa)
         twup = hwdn
         qu = -1.0
         call eaa_prof_lu_table(qu,hw,twup,irchup,ifacteaa)
         iter = iter + 1
         IF (iter .EQ. itermax) goto 20
         IF (qd-qu .GT. tolerq) THEN
c CONTINUE to decrease hwdn such that qu increases and qd decreases
            goto 10
         ENDIF
         IF (qu-qd .GT. tolerq) THEN
c overshot estimation of new hwdn; redo with 1/2 delta1
            hwdn = hwdn + delta1
            delta1 = delta1/2.
            goto 10
         ENDIF
c go it !
   20    q = amin1(qu,qd)
      ELSE IF (hw .EQ. -1.0) THEN
c  solve for headwater
         hwdn = -1.0
         call eaa_prof_lu_table(q,hwdn,tw,irchdn,ifacteaa)
         twup = hwdn
clb          hw = -1.0
         call eaa_prof_lu_table(q,hw,twup,irchup,ifacteaa)
      ELSE
c  solve for tailwater
         twup = -1.0
         call eaa_prof_lu_table(q,hw,twup,irchup,ifacteaa)
         hwdn = twup
clb          tw = -1.0
         call eaa_prof_lu_table(q,hwdn,tw,irchdn,ifacteaa)
      ENDIF
c
      RETURN
      END
```

```
      SUBROUTINE go_lu_table(go,shw,stw,istr_go)
c
c
c This SUBROUTINE RETURNs the gate opening (go); upstream stage (shw);
c or downstream stage (stw); given the values of the other two
c spillway flow variables.
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c List of Structures:
c  eaa spillway name  eaa spillway no.
c         s8                 1
c         s7                 2
c         s8new              3
c         s7new              4
c         s354               5          stagelo to lo_str_tw(imiami)
c         s351               6          stagelo to lo_str_tw(innr)
c         s352               7          stagelo to lo_str_tw(iwpb)
c
c CASE I: go must be -1.0 upon entry to this routine
c  given: downstream stage (stw), upstream stage (shw)
c         + structure number (istr_go)
c  find: gate opening (go)
c
c CASE II: shw  must be -1.0 upon entry to this routine
c  given: downstream stage (stw), gate opening (go)
c         + structure number (istr_go)
c  find: upstream stage (shw)
c
c CASE III: stw must be -1.0 upon entry to this routine
c  given: upstream stage (shw), gate opening (go)
c  find: downstream stage (stw)
c
c declare variables
c nj = 12
c    = max. number of starting downstream stages in the table
c ngo = 10
c    =  number of gate openings used in the table
c nstr = 15
c        = max. number of structures
c note: nj = ntails(istr_go)
c       ngo = n_strgos(istr_go)
c dsstg(nj,nstr), usstg(ngo,nj,nstr), strgo(ngo,nj,nstr)
c
      INTEGER istr_go
      REAL go, stw, shw, val(22)
c COMMON block for structure max. gate opening data
      include 'wmm_par.inc'
      include 'mgateop.inc'
c
c initialize variables
      nj = ntails(istr_go)
      ngo = n_strgos(istr_go)
c
      IF (stw .EQ. -1.0) THEN
c calculate go for each tw level at given hw
         DO i=1,ntails(istr_go)
            val(i) = ainterp(shw,usstg(1,i,istr_go), strgo(1,i,istr_go),
     $      ngo,0.,0.,0)
```

```
         END DO
c
c pick go "lines" where given go falls in between
         call locate(val(1),ntails(istr_go),go,ij)
         IF (ij .EQ. 0 .OR. ij .EQ. ntails(istr_go)) THEN
            ijtmp = iabs(ij-1)
            ijtmp1 = ijtmp + 1
         ELSE
            ijtmp = ij
            ijtmp1 = ij + 1
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         xgo = val(ijtmp)
         xgo1 = val(ijtmp1)
         valx = dsstg(ijtmp,istr_go)
         valx1 = dsstg(ijtmp1,istr_go)
      ELSE
c pick tw lines where given tw falls in between
         call locate(dsstg(1,istr_go), nj, stw, ij)
         IF (ij .EQ. 0 .OR. ij .EQ. nj) THEN
            ijtmp = iabs(ij-1)
            ijtmp1 = ijtmp + 1
         ELSE
            ijtmp = ij
            ijtmp1 = ij + 1
         ENDIF
c
         IF (go .EQ. -1.0) THEN
c solution for CASE I:
c call go twice for the 2 values of tw lines
            value = ainterp(shw, usstg(1,ijtmp,istr_go), strgo(1,ijtmp,
     $      istr_go), ngo, 0.,0.,0)
            value1 = ainterp(shw, usstg(1,ijtmp1,istr_go), strgo(1,
     $      ijtmp1,istr_go), ngo, 0.,0.,0)
         ELSE IF (shw .EQ. -1.0) THEN
c solution for CASE II:
c call hw twice for the 2 values of tw lines
            value = ainterp(go, strgo(1,ijtmp,istr_go), usstg(1,ijtmp,
     $      istr_go), ngo, 0.,0.,0)
            value1 = ainterp(go, strgo(1,ijtmp1,istr_go), usstg(1,
     $      ijtmp1,istr_go), ngo, 0.,0.,0)
         ELSE
            WRITE(*,*) ' Dependent variable should be defined as -1.0'
            STOP
         ENDIF
c
         xval = dsstg(ijtmp,istr_go)
         xval1 = dsstg(ijtmp1,istr_go)
      ENDIF
c
      IF (go .EQ. -1.0) THEN
c solution for CASE I:
c interpolate between the two go's
         go = value + (stw-xval)/(xval1-xval)*(value1-value)
      ELSE IF (shw .EQ. -1.0) THEN
c solution for CASE II:
c interpolate between the two shw's
```

```
         shw = value + (stw-xval)/(xval1-xval)*(value1-value)
      ELSE IF (stw .EQ. -1.0) THEN
c solution for CASE III:
c interpolate between the two stw's
         stw = valx + (go-xgo)/(xgo1-xgo)*(valx1-valx)
      ELSE
         WRITE(*,*) ' Dependent variable should be defined as -1.0'
         STOP
      ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      RETURN
      END
```

eaa_pump_caps.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: eaa_pump_caps.inc,v 1.3 2003/07/03 20:07:59 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaa_pump_caps.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      COMMON /eaa_pump_caps/ fllks6_pump,grf3amx_pump
     &, grf2amx_pump, s354_tw_pump, s351_tw_pump
     &, s354cap_pump, s351cap_pump
```

eaacap.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: eaacap.inc,v 1.4 2003/07/03 20:07:59 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaacap.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
c COMMON block for eaa canal water surface profile data
      common /eaacap/ dsstage(22,30),usstage(20,22,30),
     &eaaflow(20,22,30),ntws(30),n_eaaflows(30),nreach,is8_no_sim,
     &is7_no_sim,is6_no_sim,is5a_no_sim,is150_no_sim
```

eaacor.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: eaacor.F,v 1.18 2005/06/10 13:29:03 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/eaacor.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE ASSURES MASS BALANCE IN EVERGLADES AGRICULTURAL
C   AREA (EAA) BY APRROPRIATELY ADJUSTING WATER LEVELS IN EAA WHEN EXCESS RUNOFF
C   AND IRRIGATION REQUIREMENTS IN EXCESS OF WHAT CAN BE DELIVERED OCCUR.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE eaacorr (solmdph, supply_excess, isem_flg, jday,
     & tot_runoff_excess,add_dmndagl8,no_nonscane_irr_cells_l8)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'smallres.inc'

      DIMENSION solmdph(max_ncells), tot_runoff_excess(max_n_eaa_condt)
     $ ,DPTHCOR(max_ncells)

      REAL k, kc

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
      DO i = 1,max_ncells
         DPTHCOR(i) = 0.0
      ENDDO
c     nodebf = 23-minx(44)+1+isum(44)

C  account when runoff from EAA exceeds removal capacity

      DO index = 1,nagpts
         IF (iagflo(index,2) .NE. 40 .AND. iagflo(index,2) .NE. 42) THEN

            IF (iagflo(index,1) .GT. 0) THEN
               tot_runoff_excess(iagflo(index,2)-6) = tot_runoff_excess
     $         (iagflo(index,2)-6) + iagflo(index,1)

            ENDIF
         ENDIF
      enddo
c
      total_l8ag_dmd = amax1(dmndagl8,0.0) + add_dmndagl8
      if (total_l8ag_dmd .gt. 0.0001) THEN
         ratio_adj_n = amax1(dmndagl8,0.0)/total_l8ag_dmd
         ratio_adj_s = add_dmndagl8 / total_l8ag_dmd
      else
         ratio_adj_n = 1.0
         ratio_adj_s = 1.0
      endif
c
C  make adjustment for US Sugar ranch IF separate basin

c     IF (isem_flg .NE. 1 .AND. runmode .EQ. 'SIMUL' .AND. istaopt(10)
c    $.EQ. 1) THEN
c        IF (dmdsugh .GT. 0.0) THEN
c           iagflo(iag(isugbsn),1) = iagflo(iag(is8bsn),1)*
c    $      (NAGNODESG) / (NAGNODE(IS8BSN-6))
c           iagflo(iag(is8bsn),1) = iagflo(iag(is8bsn),1) - iagflo(iag
c    $      (isugbsn),1)
c        ENDIF
c     ENDIF
      DO 30 I = MAXY,42,-1
         DO 20 J = ISUM(I) + 1,ISUM(I+1)
            IF ((IBSN(J).LT. 7 .OR. IBSN(J) .GT. 10) .AND. IBSN(J) .NE.
     $       40 .AND. IBSN(J) .NE. 42) GO TO 20
            IAR = IBSN(J) - 6
            DO 10 INDEX = 1,NAGPTS
               IF (runmode .EQ. 'CALIB'.AND. IBSN(J) .NE. 10.AND.
     $         IBSN(J) .NE. 40) THEN
                  IF( IAGFLO(INDEX,2) .EQ. IBSN(J)) THEN
                     RCHG(J) = RCHG(J) + 86400.*IAGFLO(INDEX,1)/(GDAR
     $               *(NAGNODE(IAR)+NNODES_WETL(IAR))* (1.0 -
     $               carfact(j) - total_small_afact(j)))
                  ENDIF
               ELSE
                  IF( IAGFLO(INDEX,2) .EQ. IBSN(J)) THEN
                     IF (IBSN(J) .NE. 40 .AND. IBSN(J) .NE. 42) THEN
                        IF (iagflo(index,1) .GE. 0) THEN
                           DPTHCOR(j) = 86400.*ABS(IAGFLO(INDEX,1))
```

```
     $                    /((nagnode(iar) + nnodes_wetl(iar)) * GDAR
     $                    * (1.0 - carfact(j) - total_small_afact(j)))
                       ELSE
                          DPTHCOR(j) = 86400.*ABS(IAGFLO(INDEX,1))/
     $                    (GDAR*NAGNODE(IAR)*(1.0-carfact(j)
     $                    -total_small_afact(j)))
                       ENDIF
                    ELSEIF (IBSN(J) .EQ. 40) THEN
                       vol_cor_adj = ABS(IAGFLO(INDEX,1))*ratio_adj_n
                       DPTHCOR(j) = 86400.*vol_cor_adj / (GDAR
     $                  * NAGNODEL8 * (1.0 - carfact(j) -
     $                 total_small_afact(j)))
                    ELSE
                       IF (IAGFLO(INDEX,1) .GE. 0) THEN
                         DPTHCOR(j)=86400.*ABS(IAGFLO(INDEX,1))/(GDAR
     $                   *(NAGNODESG+NNODESSG_WETL)* (1.0-carfact(j) -
     $                   total_small_afact(j)))
                       ELSE
                         IF (NAGNODESG .GT. 0.) THEN
                           DPTHCOR(j)=86400.*ABS(IAGFLO(INDEX,1))/(GDAR
     $                     *NAGNODESG* (1.0-carfact(j) -
     $                     total_small_afact(j)))
                         ELSE
                           DPTHCOR(j)=0.0
                         ENDIF
                       ENDIF
                    ENDIF
                    IF (IAGFLO(INDEX,1) .GE. 0) THEN
                       IF (land_use_type(LUTYP(J)) .NE. 'WETLAND') THEN
                          SOLMX(J) = SOLMX(J) + DPTHCOR(j)
                          POND(J) = POND(J) + AMAX1(SOLMX(J) - SOLMDPH
     $                    (J),0.0)
                          SOLMX(J) = AMIN1(SOLMDPH(J),SOLMX(J))
                       ELSE
                          POND(J) = POND(J) + DPTHCOR(j)
                       ENDIF
                    ELSE
                       IF (land_use_type(LUTYP(J)) .NE. 'WETLAND') THEN
                          DPHSUB = AMIN1(DPTHCOR(j),POND(J))
                          POND(J) = POND(J) - DPHSUB
                          DPHDFF = DPTHCOR(j) - DPHSUB
                          ADJMX = AMIN1(SOLMX(J),DPHDFF)
                          RCHG(J) = RCHG(J) - (DPHDFF - ADJMX)
                          SOLMX(J) = SOLMX(J) - ADJMX
                       ENDIF
                    ENDIF
                 ENDIF
              ENDIF
   10       CONTINUE
C-----------------
            IF (runmode .EQ. 'SIMUL') THEN
               IF (ibsn(j) .GE. 7 .AND. ibsn(j) .le.  9 .AND.
     $         land_use_type(LUTYP(J)) .NE. 'WETLAND') THEN
                 solmx(j)=solmx(j)+stor_eaa_adj(iar)*86400./(gdar
     $           *NAGNODE(IAR)*(1.0-carfact(j)-total_small_afact(j)))
                 POND(J) = POND(J) + AMAX1(SOLMX(J) - SOLMDPH(J),0.0)
                 SOLMX(J) = AMIN1(SOLMDPH(J),SOLMX(J))
```

```
               ENDIF
            ENDIF
c           IF (land_use_type(LUTYP(J)) .NE. 'WETLAND') et_ponding(j) =
c    $      0.0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C LAND IS FLOODED
C
            IF (POND(J) .GT. 0.0 .AND. land_use_type(LUTYP(J)) .NE.
     $      'WETLAND') THEN
               SWD = POND(J)
               ITYP = LUTYP(J)
C
C     THE VALUE OF KMAX REPRESENTS THAT OF AN OPEN WATER SURFACE.
C     ALL LAND USES ARE ASSUMED TO BE OPEN WATER WHEN THE PONDING
C     DEPTH IS GREATER THAN DMAX(ARBITRARY). AS PONDING GETS
C     GREATER THIS COULD MEAN THAT THE PAN COEFFICIENT GROWS OR
C     DECREASES.
C
               K = ETK(ITYP , JDAY)
C

               IF (ITYP .EQ. 7) THEN
                  KC = K * ADJCFF(MONTH)
               ELSEIF(ITYP .EQ. 8) THEN
                  KC = K * ADJCFF(MONTH)
               ELSEIF(ITYP .EQ. 9) THEN
                  KC = K * ADJCFF(MONTH)
               ELSE
                  KC = K
               ENDIF
               etmx = KC * potet(ietzon(j))/12.
               IF (SWD .GT. OWPOND(lutyp(j))) THEN
                  ET = KMAX(lutyp(j))*POTET(ietzon(j))/12.
               ELSE
                  ET = (SWD/OWPOND(lutyp(j))*(KMAX(lutyp(j))-KC) +KC) *
     $            POTET(ietzon(j))/12.
               ENDIF
               IF (use_lec_et .NE. 0) THEN
                  et_ponding(j) = amin1(ET - ETMX,pond(j))
     $            * (1.0 - total_small_afact(j) - carfact(j))
     $            + et_ponding(j)
               ENDIF
               etm(j) = etm(j) + amin1(ET - ETMX,pond(j))
     $            * (1.0 - total_small_afact(j) - carfact(j))
               pond(j) = pond(j) - amin1(ET - ETMX,pond(j))
            ENDIF
   20    CONTINUE
   30 CONTINUE
c
c     correct for demand not met by regional system in meeting agricultural
c     demands in L-8 basin(other than sugar cane)
c
      do node = 1,isum(maxy+1)
         if (ibsn(node) .eq. 43) then
            if (frac_irr_area_t(node).gt.0.000001) THEN
             rchg(node) = rchg(node) - iagflo(iag(il8agbsn),1)
     $       * ratio_adj_s * 86400./(no_nonscane_irr_cells_l8*gdar)
```

```
            endif
         endif
      enddo
c
      DO 40 INDEX = 1,NAGPTS
   40 IAGFLO(INDEX,1) = 0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      RETURN
      END
```

environ.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c
C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: environ.inc,v 1.3 2003/07/03 20:07:59 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/environ.inc,v $
C       $Name: rel-5-5 $
-
C
C
C-----------------------------------------------------
      common /environ/
     &ngage_index_env(max_n_targ_area,max_n_loc_area,2),iwca2a_2a17,
     &inewca3a,ngrid_cells_daily_out(max_n_targ_area),
     &itarg_indx_wca(max_n_ostruc_wca,max_n_wcas,max_n_targ_struc),
     &igs333_enp,node_outflw_daily(max_n_targ_area,max_n_loc_area),
     &ngrid_cells_daily_total,ig3a28_enp,
     &stage_outflw_wca(max_n_targ_area),
     &avg_sim_wca_stg_out(max_n_targ_area),
     &no_of_targets(max_n_ostruc_wca,max_n_wcas),iwca3a_3a28,idculv2,
     &iwca3a_3a3,iwca3a_3a2,
     &itrg_byp_r_idx(max_ncnls,max_nostr_cnl,max_n_res_rec_str_flw),
     &sta34_out_to_wca3ane,
     &lec_dmnd_on_lok(max_n_wcas,max_n_conv_cnl_wca),ig1502,iewca3b,
     &sta34_out_to_s140,roten_out_to_wca3anw,roten_out_to_s140,
     &holey_out_to_nwc,holey_out_to_ds,s10eevq,s10ergq,s355_flow_targ,
     &sta34_out_to_wca3anw,irotdnod1,irotdnod2,
     &stage_ref_env(max_n_wcas,max_n_conv_cnl_wca),
     &nmpts_wca_floor(max_n_wcas,max_n_minfllvl_loc),
     &icell_no_floor(max_n_wcas,max_n_minfllvl_loc,7),
     &wca_subarea_floor(max_n_wcas,max_n_minfllvl_loc),
     &no_of_subareas_floor(max_n_wcas),idculv1,
     &min_consec_days_bel_strtstg,
     &no_days_below_floor(max_n_wcas,max_n_minfllvl_loc),
     &total_outflow_from_roten,outflow_from_roten_to_west,
     &outflow_from_roten_to_wca,frac_sem_cyp,iroten_canal_no,
     &capc_new_pump_to_west_frm_roten,dischg_to_s8_frm_sta34,
     &dischg_to_s7_frm_sta34,outflow_from_roten_to_semcyp,
     &inflow_to_sta6_frm_unit1,outflow_from_sta6,node_roten2,
     &outflow_from_sta6_to_wca,outflow_from_sta6_to_semcyp,
```

```
     &outflow_from_roten_thru_s8,runoff_thru_s150,total_outf_from_holy,
     &iopt_to_deiver_bel_targ
     &(max_ncnls,max_nostr_cnl,max_n_res_rec_str_flw),
     &i_res_inf_struc_indx
     &(max_ncnls,max_nostr_cnl,max_n_res_rec_str_flw),
     &res_struc_inflow_name(max_ncnls,max_nostr_cnl),s12_flow_targ,
     &s333_flow_targ,frac_reg_flw_nesrs_s333,frac_reg_flw_nesrs_s355,
     &nodnwc3a,inwcwca3a,rlok_flow_to_enp,rlok_flow_to_enp_s150,
     &rlok_flow_to_enp_s354,no_of_days_wk_wcadel(max_n_wcas,2),
     &frac_wcaws_del(max_n_wcas,2),
     &idays_of_wk_wcadel_indx(max_n_wcas,7,2),
     &frac_del_frm_wca(max_n_wcas)
C
       real inflow_to_sta6_frm_unit1
       character res_struc_inflow_name*6
```

et.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: et.inc,v 1.5 2003/07/03 20:08:00 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/et.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      common /ET/ ETK(max_n_lu_type,366),KMAX(max_n_lu_type),
     &OWPOND(max_n_lu_type),MOREFET(12),LOKMORN(12),POTET(max_ncells),
     &pet_weight(max_ncells,20),DRYETVOL,LITETVOL,OPNETVOL,
     &et_cal_adj_fact(max_n_lu_type,12),
     &owpond2(max_ncells)
      REAL MOREFET,LOKMORN,KMAX,LITETVOL
C-----------------
```

etcomp.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: etcomp.F,v 1.11 2005/06/10 13:29:04 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/etcomp.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE CALCULATES EVAPOTRANSPIRATION FROM THE
C   SATURATED ZONE FOR EACH GRID CELL
C
C   VARIABLE LIST ; EVAPOTRANSPIRATION CALCULATION
C
C   D,SD       LINEAR REDUCTION TERMS FOR ET BASED ON WATER TABLE POS.
C   DRZ        VALUE OF GWD BELOW WHICH NO ET OCCURS
C   ET         FINAL VALUE RETURNED BY SUBROUTINE (FT)
C   GWD        DISTANCE FROM LAND SURFACE TO WATER TABLE
C   SRZ        THICKNESS OF SHALLOW ROOT ZONE
C              (NO ET REDUCTION WHEN GWD < OR = SRZ )
C
C-----------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE etcomp (ietopt, et, node, ityp, jday, depth_of_deficit,
     & stage_res, et_outside_res, iresnum, no_in_cell,
     & sfactor_small_res, ells_in_res,iet_accnt)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
```

```
      INCLUDE 'resadj.inc'

      DIMENSION ells_in_res(max_ncells,max_n_res_in_cell),
     & stage_res(max_ncells,max_n_res_in_cell)

      REAL k, kc

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  THE VALUE OF KMAX REPRESENTS THAT OF AN OPEN WATER SURFACE.  ALL LAND
C  USES ARE ASSUMED TO BE OPEN WATER WHEN THE PONDING DEPTH IS GREATER
C  THAN DMAX(ARBITRARY). AS PONDING GETS GREATER THIS COULD MEAN THAT THE
C  PAN COEFFICIENT GROWS OR DECREASES.

      IF (ietopt .ge. 0) THEN
      K = ETK(ITYP , JDAY)

      IF ( (IBSN(NODE).GE.7 .AND. IBSN(NODE).LE.10 .AND.
     &   land_use_type(LUTYP(NODE)).NE.'WETLAND') .OR. IBSN(NODE)
     &   .EQ. 40 .OR. IBSN(NODE) .EQ. 42 .OR. (LUTYP(NODE).GE. 8
     &   .AND. LUTYP(NODE) .LE. 9)) THEN
         FAC = ADJCFF(MONTH)
      ELSE
         FAC = 1.0
      ENDIF
c     print *,IBSN(NODE),FAC,NODE,LUTYP(NODE),'  CVCVCCV'
C
      IF (ITYP.EQ. 7) THEN
         KC = K * FAC * et_cal_adj_fact(ITYP,MONTH)
      ELSEIF(ITYP .EQ. 8) THEN
         KC = K * FAC * et_cal_adj_fact(ITYP,MONTH)
      ELSEIF(ITYP .EQ. 9) THEN
         KC = K * FAC * et_cal_adj_fact(ITYP,MONTH)
      ELSE
         KC = K * et_cal_adj_fact(ITYP,MONTH)
      ENDIF
C
       GWD = ELLS(NODE) - (H(NODE) + POND(NODE))
       et_cnl=kmax(lutyp(node))*potet(ietzon(node))/12.0
C
      IF (GWD .GT. 0.0) THEN
C
C LAND IS NOT FLOODED
C
         IF (GWD .GE. DRZ(ITYP)) THEN
           ET = 0.0
         ELSEIF (GWD .GT. SRZ(ITYP)) THEN
           ET = KC*POTET(ietzon(node))/12.*(DRZ(ITYP)-GWD)/(DRZ(ITYP)
     &                - SRZ(ITYP))
         ELSE
           ET = KC*POTET(ietzon(node))/12.
         ENDIF
         if (use_lec_et.ne.0 .and. iet_accnt .eq. 1) then
            et_sat(node) = ET * (1.0-carfact(node)-sfactor_small_res)
     &      + et_sat(node)
            et_ponding(node) = et_cnl * carfact(node)
     &      + et_ponding(node)
```

```
         endif
      ELSE
C
C LAND IS FLOODED
C
         SWD = POND(NODE)
         IF (SWD .GT. OWPOND2(node)) THEN
           ET = KMAX(lutyp(node))*POTET(ietzon(node))/12.
         ELSE
           ET = (SWD/OWPOND2(node)*(KMAX(lutyp(node))-KC)
     $              +KC) * POTET(ietzon(node))/12.
         ENDIF
c
         ET_outside_res = ET
         DO ir = 1,noresincell(node)
           ia = ires_index_in_cell(node,ir)
           IF ((sfactor(ia) .GE. sfactmin .OR. ires_small_sim
     $     (ia) .eq. 'NO ') .and. sfactor(ia).lt. 1.0) THEN
             depth_of_deficit1 = abs(depth_of_deficit)
             depth_water_in_res = amax1(stage_res(node,no_in_cell)
     $       -ells(node),0.0)
            IF (depth_water_in_res .GT. OWPOND2(node)) THEN
              ET_in_res = KMAX(lutyp(node))*POTET(ietzon(node))/12.
            ELSE
              ET_in_res = (depth_water_in_res/OWPOND2(node)
     $            *(KMAX(lutyp(node))-KC) +KC) * POTET(ietzon(node))/12.
            ENDIF
             IF (depth_of_deficit .GE. DRZ(ITYP)) THEN
               ET_outside_res = 0.0
             ELSEIF (depth_of_deficit .GT. SRZ(ITYP)) THEN
               ET_outside_res = KC*POTET(ietzon(node))/12.*(DRZ(ITYP)
     &           - depth_of_deficit) / (DRZ(ITYP) - SRZ(ITYP))
             ELSE
               depth_water_o_res = depth_of_deficit1
               ET_outside_res = (depth_water_o_res/OWPOND2(node)
     &          *(KMAX(lutyp(node))-KC) +KC) * POTET(ietzon(node))
     &          /12.
             ENDIF
             ET = ET_outside_res * (1.0-sfactor(ia))
     $              + ET_in_res * sfactor(ia)
           endif
         enddo

         if (use_lec_et.ne.0 .and. iet_accnt .eq. 1) then
            etwponding = ET
            etsatur = 0.0
            if(etwponding.gt.pond(node))then
               etsatur = etwponding - pond(node)
               etwponding = pond(node)
            endif
            et_ponding(node) = etwponding*(1.0-sfactor_small_res
     $      -carfact(node))  + et_cnl*carfact(node) + et_ponding(node)
            et_sat(node) = etsatur*(1.0-sfactor_small_res
     $      -carfact(node)) + et_sat(node)
         endif
      ENDIF
      ELSE
```

```
        K = ETK(ITYP , JDAY)
        KC = K*et_cal_adj_fact(ITYP,MONTH)
        depth_water_in_res = amax1(stage_res(node,no_in_cell)
     $   - ells_in_res(node,no_in_cell),0.0)
        IF (depth_water_in_res .GT. 0.0) THEN
          IF (depth_water_in_res .GT. OWPOND2(node)) THEN
            ET_in_res = KMAX(lutyp(node))*POTET(ietzon(node))/12.
          ELSE
            ET_in_res = (depth_water_in_res/OWPOND2(node)
     $          *(KMAX(lutyp(node))-KC) +KC) * POTET(ietzon(node))/12.
          ENDIF
          if (use_lec_et.ne.0 .and. iet_accnt .eq. 1) then
             et_ponding_res = ET_in_res
             if(et_ponding_res.gt.depth_water_in_res) then
                et_sat(node) = (et_ponding_res - depth_water_in_res)
     $          *sfactor(iresnum) + et_sat(node)
                et_ponding(node) = depth_water_in_res
     $          *sfactor(iresnum) + et_ponding(node)
             else
                et_ponding(node) = et_ponding_res
     $          *sfactor(iresnum) + et_ponding(node)
             endif
c            et_pond_sep_res(node) = et_ponding(node)
          endif
        ELSE
          GWD = ells_in_res(node,no_in_cell)-stage_res(node,no_in_cell)
          IF (GWD .GE. DRZ(ITYP)) THEN
            ET_in_res = 0.0
          ELSEIF (GWD .GT. SRZ(ITYP)) THEN
            ET_in_res = KC*POTET(ietzon(node))/12.*(DRZ(ITYP)-GWD)
     &              /(DRZ(ITYP) - SRZ(ITYP))
          ELSE
            ET_in_res = KC*POTET(ietzon(node))/12.
          ENDIF
          if (use_lec_et.ne.0 .and. iet_accnt .eq. 1) then
            et_sat(node) = et_sat(node) + ET_in_res*sfactor(iresnum)
          endif
        ENDIF
        ET =  ET_in_res
       ENDIF

C
       RETURN
       END
```

excadjfresstg.F

```
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: excadjfresstg.F,v 1.7 2005/06/10 13:29:04 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/excadjfresstg.F,v $
C   $Name: rel-5-5 $
C
C  This subroutine calculates depth of excess water used to adjust
C  estimate of reservoir stage for reservoirs whose actual
c  area is different from grid cell(s) reservoir passes
c  through and not treated as separate entity from grid cell(s)
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE excadjfresstg (sumrain, rain, dpthgw_for_rain,
     & total_afact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'

      DIMENSION dpthgw_for_rain(max_ncells), rain(max_ncells),
     & sumrain(max_ncells), total_afact(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  calculate depth of excess water to be used to adjust
C  estimate of reservoir stage for reservoirs whose actual
c  area is different from grid cell(s) reservoir passes
c  through and not treated as separate entity from grid cell(s).

      DO node=1,isum(maxy)
        head=h(node)+pond(node)
        DO ir=1,noresincell(node)
          ia = ires_index_in_cell(node,ir)
          IF (sfactor(ia) .GT. sfactmin .OR. ires_small_sim
```

```
     &    (ia) .EQ. 'NO ') THEN
            IF (head.le.ells(node)) THEN
              afact=1.0
            ELSE
              afact=total_afact(node)
            END IF
            IF (head.gt.ells(node)) THEN
              adjust_for_ovlf=sum_chg_dpth_ovl_flw(node)
              IF (sfactor(ia).le.1.0) THEN
                sgn=1
              ELSE
                sgn=-1
              END IF
              term1_for_adjust_area=(sumet_outside_res(node)
     &         +seep_loss_res(node)-sumrain(node)/12.0
     &         -sum_chg_depth_levee_spg(node))
     &         *((1.0-afact)/afact)
              term2_for_adjust_area=(outf_fc_frm_res_to_cell(node)+
     &         outf_seep_loc_to_res_cell(node)+
     &         daily_outf_fc_frm_res_to_cell(node)+adjust_for_ovlf)/
     &         afact
              IF (sgn*term1_for_adjust_area.gt.term2_for_adjust_area)
     &         THEN
                daily_excess_for_adj=(amax1(rain(node)/12.-
     &           dpthgw_for_rain(node),0.0)-daily_seep_loss_res(node)-
     &           daily_et(node)+chg_depth_levee_spg(node))
     &           *((1.0-afact)/afact)
     &           +daily_outf_fc_frm_res_to_cell(node)/afact+
     &           daily_chg_dpth_ovl_vol(node)/afact
                IF (head.gt.ells(node)) THEN
                  daily_excess_adjust(node)=daily_excess_for_adj+
     &             daily_excess_adjust(node)
!                if( resname(ia).eq.'ENPBUF') print *,
!    $             daily_excess_for_adj,daily_excess_adjust(node)
!    $                ,'  DEA  ',rain(node)/12.,dpthgw_for_rain(node)
!    $                ,daily_seep_loss_res(node),daily_ET(node)
!    $                ,daily_outf_fc_frm_res_to_cell(node)
                END IF
                IF (sgn*daily_excess_adjust(node).lt.sgn*0.0001) THEN
                  daily_excess_adjust(node)=0.0
                END IF
              ELSE
                daily_excess_adjust(node)=0.0
              END IF
!             if (resname(ia).eq.'ENPBUF')
!    $        print *,HEAD,term1_for_adjust_area
!    $            ,term2_for_adjust_area,daily_excess_adjust(node)
!    $            ,'   DAILY EXCESS ADJUST'
            ELSE
              daily_excess_adjust(node)=0.0
            END IF
          ENDIF
        END DO
      END DO
      RETURN
      END
```

fin_go.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: fin_go.inc,v 1.3 2003/07/03 20:08:00 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/fin_go.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
c COMMON block for actual gate openings
      COMMON /fin_go/ go_s8,go_s7,go_s8new,go_s7new
     &, go_s354,go_s351,go_s352
```

final_output.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: final_output.F,v 1.11 2005/06/10 13:29:04 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/final_output.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE OUTPUTS 1) MEAN NUMBER OF DAYS PER YEAR THE LIMIT
C   FOR OVERLAND FLOW FROM EACH GRID CELL  IS REACHED TO AN ASCII
C   FILE, AND 2) CELL WATER LEVELS ,CANAL STAGES,AND LAKE OKEECHOBEE STAGE
C   TO RESTART FILE.  THIS IS DONE AT END OF LAST DAY OF SIMULATION.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE final_output (stagelo)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'ovfllim.inc'

      CHARACTER*200 filename

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   OUTPUT mean number of days per year volume limit for overland flow
C      is reached

      IF (LP(11)) THEN
         DO iyear = 1,nyear
            WRITE(69,10) IFYR+IYEAR-1
            DO it = 1,2
               if (it .EQ. 1) THEN
                  WRITE(69,20)
```

```
               ELSE
                  WRITE(69,30)
               ENDIF
               DO I = MAXY,1,-1
                  LL = (MINX(I) - 1) * 5 + 1
                  WRITE(69,'(A,50(i5,1x))') B(1:LL),
     $            (ndays_ovlflo_vol_limit (node,iyear,it),NODE = ISUM(I)
     $            + 1, ISUM(I + 1))
               END DO
               DO I = MAXY,1,-1
                  LL = (MINX(I) - 1) * 5 + 1
                  WRITE(69,'(A,50(f6.2,1x))') B(1:LL),
     $            (ratio_excess_ovlflo (node,iyear,it),NODE = ISUM(I) +
     $            1, ISUM(I + 1))
               END DO
            END DO
         END DO
      END IF
C
C                           WRITE RESTART TAPE IF LP 2 IS SET
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF( LP(2) ) THEN
         FILENAME = DATADIR (1:LENGTH(DATADIR)) // '/restart_output'
         OPEN(55,FILE = FILENAME)
         DO 40 I = MAXY,1,-1
            LL = (MINX(I) - 1) * 5 + 1
            WRITE(55,80) B(1:LL),(H(NODE),NODE=ISUM(I)+1,ISUM(I+1))
   40    CONTINUE
         DO 50 I = MAXY,1,-1
            LL = (MINX(I) - 1) * 5 + 1
            WRITE(55,80)B(1:LL),(POND(NODE),NODE=ISUM(I)+1,ISUM(I+1))
   50    CONTINUE
         WRITE(55,60) STAGELO
         IF (runmode .EQ. 'CALIB') THEN
            WRITE(55,*) NCH
         ENDIF
         WRITE(55,'(15F6.2)') (CHDEP(IN),IN = 1,NCH)
         IF (type_flow_across_ttrail.NE.'MINDEL'.AND..NOT. opt_ft_input)
     $   WRITE(55,70) FRFP,FRF
      ENDIF
      RETURN
   10 FORMAT(2(/),5X,'YEAR  =  ',I5,2(/))
   20 FORMAT(5x,'EAST_WEST DIRECTION',2(/))
   30 FORMAT(5x,'NORTH_SOUTH DIRECTION',2(/))
   60 FORMAT(1X,F6.2)
   70 FORMAT(2F10.0)
   80 FORMAT(A,42F5.1)
      END
```

flow_to_stas.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: flow_to_stas.F,v 1.18 2005/06/10 13:29:05 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/flow_to_stas.F,v $
C   $Name: rel-5-5 $
C
C   the purpose of this subroutine is to route runoff
C   and environmental water supply volumes from appropriate areas
C   to corresponding stormwater treatment area (STA) as part of
C   Everglades Construction Project (ECP).
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE flow_to_stas (runoff, dmnd_lec, flow_through, istopt,
     & dmds236, s236so, volwtr, dellec, ws_from_lo_s6, ifacteaa,
     & kfloref, runoff_south, flow_from_eaa_lok_s8,
     & flow_from_eaa_lok_s7, ws_frm_lok_to_semcyp,
     & avail_cap_for_sta34_nnr, avail_cap_for_sta34_miami,
     & ws_to_sta34_via_nnrc, add_conv_red_miami, ws_to_sta34_frm_lok,
     & ws_frm_lok_to_holy, frac_env_met, dmnd_lec_sa2_s7,
     & dmnd_lec_sa3_s150,pctmet, climate_index,iday_lok_in_ssm,
     & bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a,
     & rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'wcaot.inc'
```

```
      INCLUDE 'lakwca.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'eaa_caps.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'climvar.inc'
      INCLUDE 'flw298dso.inc'
      INCLUDE 'loktarg.inc'

      DIMENSION climate_index(12), runoff_south(max_n_eaa_condt),
     & volwtr(max_n_res),LWSLECSTA(max_n_eaa_condt),
     & res_outflow_to_basin(max_n_res,max_nostr_res),
     & temp_avail_stor(max_n_res),flo_eaa_cnl(max_n_eaa_condt),
     & rinf_struc_cap(max_n_eaa_condt),
     & add_conv_red(max_n_eaa_condt)

      LOGICAL l3a28_only,LWSLECSTA

      REAL kfloref(max_n_eaa_condt), nnrc_cap_lower_grav_s150
      CHARACTER bmp_ssm_opt*3
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (DRYSEAS) THEN
         iseas = 1
      ELSE
         iseas = 2
      ENDIF

      total_inflow = runoff + flow_through
      total_inflow_unadj = total_inflow
      flow_through_unadj = flow_through

C  runoff from s-8 basin into sta3 and sta4

      IF (istopt .EQ. 1) THEN
         avail_cap_for_sta34_miami= amin1(g372cap*pctwma (indxsta
     $   (is8bsn-6))-env_ws_from_res_thru_g372 ,volwtr(indxsta(is8bsn-6)
     $   ))
         pumpcap_res(indxsta(is8bsn-6)+1) = g372cap*pctwma (indxsta
     $   (is8bsn-6)) - env_ws_from_res_thru_g372
c
c  recompute conveyance capac for neutral case based on HEC-2 lookup
c  tables when pumping into STA3&4 from Miami Canal is occurring
c
         istype = 1
         ireach = 5
         ctw = pmp_int_new(is8bsn-6)
         IF (.not. LFORPUMP(is8bsn-6)) THEN
           istr_eaa = 5
           chw = stagelo
           delta0 = (chw-ctw)/25.
           call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo,delta0,
     $     1.0,100,qcanal,strucap,s354_tw_temp ,ifacteaa)
```

```
            IF (s354_tw_temp .GT. rmaxstg(is8bsn-6)) THEN
               s354_tw_temp = rmaxstg(is8bsn-6)
               qcanal = -1.0
               IF (s354_tw_temp .GT. ctw) THEN
                  call eaa_prof_lu_table(qcanal,s354_tw_temp,ctw, ireach,
     $            ifacteaa)
               ELSE
                  qcanal = 0.0
               ENDIF
            ENDIF
            s354cap_temp = -1.0
            call eaaspill(s354cap_temp,stagelo,s354_tw_temp,5,ifacteaa)
         ELSE
            s354_tw_temp = amin1(rmaxstg(is8bsn-6),stagelo
     $      + rmax_lift(is8bsn-6))
            s354cap_temp = -1.0
            call eaaspill(s354cap_temp,stagelo,s354_tw_temp,9,ifacteaa)
            qcanal = s354cap_temp
            s354_tw_temp_max = s354_tw_temp
            s354_tw_temp = -1.0
            call eaa_prof_lu_table(qcanal,s354_tw_temp,ctw, ireach,
     $      ifacteaa)
            IF (s354_tw_temp .GT. s354_tw_temp_max) THEN
               s354_tw_temp = s354_tw_temp_max
               qcanal = -1.0
               call eaa_prof_lu_table(qcanal,s354_tw_temp,ctw, ireach,
     $         ifacteaa)
            ENDIF
         ENDIF
         grf3amx_to_sta = amin1(qcanal,s354cap_temp)
         grf3amx_prescap_to_sta = grf3amx_to_sta / amax1
     $   (rcpfacteaa_canal(is8bsn-6),1.0)
         ncount(indxsta(is8bsn-6)) = ncount(indxsta(is8bsn-6)) + 1
         oper_capac = cnlcap(imiami) - (1.0-pct_des(imiami,iseas))
     $   *cnlcap(imiami)/rcpfacteaa(is8bsn-6)
         ntarg = 0
         DO itarg = 1,no_targ_loc(is8bsn-6)
            IF (lwslo(itarg_area(itarg,is8bsn-6))) THEN
               ntarg = ntarg + 1
            ENDIF
         END DO
         l3a28_only = ntarg .EQ. 1 .AND. lwslo(iwca3a_3a28)
         IF (ifacteaa .EQ. 1) THEN
            flow_thru_prescap = amax1(grf3amx_prescap - kfloref
     $      (is8bsn-6),0.0)
            flow_thru_prescap = amin1(flow_thru_prescap,flow_through)
            total_inflow_prescap = runoff + flow_thru_prescap
         ELSE
            total_inflow_prescap = total_inflow
         ENDIF
         total_inflow_prescap_unadj = total_inflow_prescap
c        print *,ntarg,flow_thru_prescap,runoff,flow_through
c    $     ,total_inflow,total_inflow_prescap,'  HJKL'
         IF (ntarg .gt. 0) THEN
            if(ifacteaa .EQ. 1) flow_thru_prescap_to_sta=
     $      flow_thru_prescap + amax1(grf3amx_prescap_to_sta -
     $      grf3amx_prescap,0.0)
```

```
            capac_incrse = amax1(grf3amx_to_sta-grf3amx,0.0)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            flow_through_to_sta = flow_through + capac_incrse
         ELSE
            if(ifacteaa.EQ.1) flow_thru_prescap_to_sta=
     $      flow_thru_prescap
            flow_through_to_sta = flow_through
         ENDIF
         total_flow_env = runoff + flow_through_to_sta
         IF ((total_inflow_prescap .GT. dmnd_lec .OR. (ifacteaa .EQ. 1
     $   .AND. total_inflow .GT. dmnd_lec)).OR.((((LWSLO_floor(iwca3a)
     $   .AND. LWSLECSTA(is8bsn-6).AND.floor_grid_cell_opt.eq.'MARSH')
     $   .OR. rmin_stg_wca3a .gt.rmnstg(iwca3a,iconvey_canal_no_wca
     $   (iwca3a))+0.001)).AND. iopt_ws_lec_to_sta_wca_abv_flr .eq. 1)
     $   .OR. flow_to_wca_prior .EQ.'LEC') THEN
            s354cap = s354cap_temp
            s354_tw = s354_tw_temp
            s354mx = s354cap
            IF (.not. LFORPUMP(is8bsn-6)) then
               IF (stagelo .LE. pmp_int_new(is8bsn-6)) THEN
                  total_inflow = runoff
                  flow_through = 0.0
                  total_inflow_prescap = runoff + flow_through
                  flow_thru_prescap = flow_through
               ENDIF
            ENDIF
            IF (.NOT. DRYSEAS .AND. nint(runoff) .GT. 0) THEN
               rmax_to_glades = amin1(grf3amx_to_sta,oper_capac) -
     $         kfloref(is8bsn-6)
            ELSE
               rmax_to_glades = amin1(grf3amx_to_sta-kfloref(is8bsn-6) ,
     $         oper_capac)
            ENDIF
            rmax_to_glades = amax1(rmax_to_glades,0.0)
            rmax_to_glades = amin1(rmax_to_glades ,
     $      avail_cap_for_sta34_miami)
c           print *,rmax_to_glades,grf3amx_to_sta,oper_capac
c    $      ,kfloref(is8bsn-6),'  REMTOGLDS  ',avail_cap_for_sta34_miami
c
            flow_thru_prescap_to_sta = flow_thru_prescap + amax1
     $      (grf3amx_prescap_to_sta - grf3amx_prescap,0.0)
c
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               IF (stagelo .GT. pmp_int_new(is8bsn-6)) THEN
                  IF (ntarg .GT. 0) THEN
                     avail_flow_to_lec = frac_lec(is8bsn-6) *
     $               total_inflow
                  ELSE
                     avail_flow_to_lec = total_inflow
                  ENDIF
                  flow_to_lec = amin1(avail_flow_to_lec,dmnd_lec)
                  rem_to_glades = amax1(total_flow_env - flow_to_lec,
     $            0.0)
               ELSE
                  IF (ntarg .GT. 0) THEN
                     avail_flow_to_lec = flow_through_unadj + frac_lec
     $               (is8bsn-6) * total_inflow
```

```
                     rem_to_glades = total_inflow-amin1(frac_lec
     $               (is8bsn-6) *total_inflow,dmnd_lec)
                  ELSE
                     avail_flow_to_lec = flow_through_unadj +
     $               total_inflow
                     rem_to_glades = total_inflow - amin1(total_inflow ,
     $               dmnd_lec)
                  ENDIF
                  flow_to_lec = amin1(avail_flow_to_lec,dmnd_lec)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               ENDIF
               rem_to_glades = amin1(rem_to_glades ,
     $         avail_cap_for_sta34_miami)
               flow_thru_to_glades_bfop = amax1(rem_to_glades-runoff,
     $         0.0)
               flow_thru_to_glades = amin1(flow_thru_to_glades_bfop ,
     $         rmax_to_glades)
               total_inflow = amin1(runoff,rem_to_glades) +
     $         flow_thru_to_glades
               flow_through = flow_through_to_sta
     $         -(flow_thru_to_glades_bfop - flow_thru_to_glades)
               miamic_cap_upper_grav = grf3amx_to_sta
               miamic_cap_upper_pump = grf3amx_to_sta
            ELSE
               IF (nasr_to_lok(1) .EQ. 0) THEN
                  call bpts(iday,month,10.0,1,n_fc_zones+4,1,
     $            stgref_asr_rec)
                  IF (stagelo .lt. stgref_asr_rec .AND.(climate_index
     $            (month) .LT. 1.5 .OR. ((itrib_hydro_cond_opt .EQ. 1
     $            .AND. trib_rf_et .LT. -1.0 .AND.
     $            s65e_runff_wkly .LT. 1500.) .OR.
     $            itrib_hydro_cond_opt .EQ. 0)) .AND. DRYSEAS) THEN
                     frac_env_deliv = 0.00
                  ELSE
                     frac_env_deliv = 1.00
                  ENDIF
               ELSE
                  frac_env_deliv = 1.00
               ENDIF
               frac_env_deliv = amin1(frac_env_deliv,frac_env_met)
               IF (l3a28_only) THEN
                  depth_below_targ_3a28 = stage_import_wca(iwca3a_3a28)
     $            - offset(iwca3a_3a28,1)-avg_sim_wca_stg(iwca3a_3a28)
                  depth_below_targ_3a28 = amax1(depth_below_targ_3a28,
     $            0.0)
                  IF (depth_below_targ_3a28 .LT. 0.1) THEN
                     frac_rem_capac_miami = 10.0*depth_below_targ_3a28
     $               *frac_rem_capac(is8bsn-6)
                  ELSE
                     frac_rem_capac_miami = frac_rem_capac(is8bsn-6)
                  ENDIF
               ELSE
                  frac_rem_capac_miami = frac_rem_capac(is8bsn-6)
               ENDIF
               frac_rem_capac_miami = frac_env_deliv
     $         *frac_rem_capac_miami
               IF (ifacteaa.EQ. 1 .AND. total_inflow_unadj .GT.
```

```
     $         dmnd_lec .AND. total_inflow_prescap_unadj .LE. dmnd_lec)
     $         THEN
                  flow_thru_glades = amax1(flow_through_to_sta
     $            -flow_thru_prescap,0.0)
                  rmax_to_glades = amin1(rmax_to_glades,
     $            flow_thru_glades) * frac_rem_capac_miami
                  total_inflow = amin1(rmax_to_glades ,
     $            avail_cap_for_sta34_miami)
               ELSE
                  flow_through = amin1(flow_through_to_sta,
     $            rmax_to_glades) * frac_rem_capac_miami
                  total_inflow = runoff + flow_through
                  IF (runoff .LE. avail_cap_for_sta34_miami) THEN
                     flow_through = flow_through - (total_inflow - amin1
     $               (total_inflow,avail_cap_for_sta34_miami))
                     total_inflow = amin1(total_inflow ,
     $               avail_cap_for_sta34_miami)
                  ELSE
                     total_inflow = runoff
                     flow_through = 0.0
                  ENDIF
               ENDIF
               miamic_cap_upper_grav = grf3amx_to_sta
               miamic_cap_upper_pump = grf3amx_to_sta
            ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            rmax_inflow = amin1(total_inflow,g372cap*pctwma (indxsta
     $      (is8bsn-6))- env_ws_from_res_thru_g372 ,volwtr(indxsta
     $      (is8bsn-6)))
c           print *,rmax_inflow,total_inflow,g372cap
c    $      ,pctwma (indxsta(is8bsn-6)),env_ws_from_res_thru_g372
c    $      ,volwtr(indxsta(is8bsn-6)),'  JKLJKL  '
c    $      ,frac_rem_capac_miami,frac_env_deliv,frac_env_met
c    $      ,runoff,flow_through,l3a28_only,dmnd_lec
c    $      ,flow_through_to_sta,flow_thru_glades
            qinflw(indxsta(is8bsn-6),ncount(indxsta(is8bsn-6))) = qinflw
     $      (indxsta(is8bsn-6),ncount(indxsta(is8bsn-6))) + rmax_inflow

c
c    compute excess runoff from s-8 basin sta34 could not take
c
            rem_runoff_over_des_news8 = amax1(total_inflow - g372cap *
     $      pctwma(indxsta(is8bsn-6))+env_ws_from_res_thru_g372,0.0)
            rem_runoff_over_stor_sta34 = amax1(total_inflow - volwtr
     $      (indxsta(is8bsn-6)),0.0)
            pumpcap_res(indxsta(is8bsn-6)+1) = amax1(g372cap*pctwma
     $      (indxsta(is8bsn-6))- rmax_inflow -
     $      env_ws_from_res_thru_g372,0.0)

c
c    compute remaining available storage in sta34
c
            volwtr(indxsta(is8bsn-6)) = volwtr(indxsta(is8bsn-6)) -
     $      qinflw(indxsta(is8bsn-6),ncount(indxsta(is8bsn-6)))
            flow_s8new_to_sta = rmax_inflow
            rmiami_runoff_to_sta = amin1(rmax_inflow,runoff)
            IF (ifacteaa.EQ. 1 .AND. total_inflow_unadj .GT. dmnd_lec
```

```
     $      .AND. total_inflow_prescap_unadj .LE.dmnd_lec)
     $      rmiami_runoff_to_sta = 0.0
c
c    route excess flow south to wca-3a
c
            flow_lower_reach_max = -1.0
            ireach = 8
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               total_flow_s8new = total_inflow + flow_to_lec
     $          - amax1(flow_to_lec - runoff,0.0) * (1.0
     $          - frac_del_frm_wca(iwca3a))
            ELSE
               IF (.not. LFORPUMP(is8bsn-6)) THEN
                  IF (stagelo .LE. pmp_int_new(is8bsn-6)) THEN
                     add_flow_thru = amin1(dmnd_lec,flow_through_unadj)
                  ELSE
                     add_flow_thru = 0.0
                  ENDIF
               ELSE
                  add_flow_thru = 0.0
               ENDIF
               IF (ifacteaa.EQ. 1 .AND. total_inflow_unadj .GT.
     $         dmnd_lec .AND. total_inflow_prescap_unadj .LE. dmnd_lec)
     $             THEN
                  total_flow_s8new = total_inflow +
     $            total_inflow_prescap + add_flow_thru
     $            * frac_del_frm_wca(iwca3a)
     $            - amax1(total_inflow_prescap - runoff,0.0)
     $            * (1.0 - frac_del_frm_wca(iwca3a))
               ELSE
                  total_flow_s8new = total_inflow + add_flow_thru
     $            * frac_del_frm_wca(iwca3a)
               ENDIF
            ENDIF
            IF (eaa_conv_opt_s8 .NE. 'PUMP' .AND. runoff .LE.
     $      avail_cap_for_sta34_miami) THEN
c   water supply releases by gravity only
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               total_flow_upper_reach = total_flow_s8new +
     $         dmnd_miami_basin + add_conv_red_miami +
     $         ws_to_sta34_frm_lok + ws_frm_lok_to_semcyp +
     $         ws_frm_lok_to_holy
               IF (total_flow_upper_reach .GT. miamic_cap_upper_grav)
     $         THEN
                  fcuhw = -1.0
                  ireach = 5
                  IF (flow_s8new_to_sta .GT. 0.0) THEN
                     s8_new_hw = pmp_int_new(is8bsn-6)
                  ELSE
                     s8_new_hw = amax1(s8_new_hw,pmp_int_new(is8bsn-6))
                  ENDIF
                  call eaa_prof_lu_table(total_flow_upper_reach,fcuhw ,
     $            s8_new_hw,ireach,ifacteaa)
                  IF (.not. LFORPUMP(is8bsn-6)) THEN
                     fcuhw = amin1(fcuhw,rmaxstg(is8bsn-6))
                  ELSE
                     fcuhw = amin1(fcuhw,rmaxstg(is8bsn-6),
```

```
     $               stagelo+rmax_lift(is8bsn-6))
                  ENDIF
                  s354_tw = fcuhw
                  miamic_cap_upper_grav = -1.0
                  IF (s354_tw.GT.s8_new_hw) THEN
                     call eaa_prof_lu_table(miamic_cap_upper_grav,
     $               s354_tw, s8_new_hw,ireach,ifacteaa)
                  ELSE
                     miamic_cap_upper_grav = 0.0
                  ENDIF
                  s354cap = -1.0
                  IF (.not. LFORPUMP(is8bsn-6)) THEN
                     call eaaspill(s354cap,stagelo,s354_tw,5,ifacteaa)
                  ELSE
                     call eaaspill(s354cap,stagelo,s354_tw,9,ifacteaa)
                  ENDIF
               ELSE
                  IF (total_flow_upper_reach .GT. 0.0) THEN
                     IF (flow_s8new_to_sta .GT. 0.0) THEN
                        fcutw = -1.0
                        call eaa_prof_lu_table(total_flow_upper_reach,
     $                  s354_tw ,fcutw,5,ifacteaa)
                        s8_new_hw = amax1(fcutw,pmp_int_new(is8bsn-6))
                        IF (fcutw .LT. pmp_int_new(is8bsn-6)) THEN
                           fcuhw = -1.0
                           call eaa_prof_lu_table
     $                     (total_flow_upper_reach ,fcuhw,pmp_int_new
     $                     (is8bsn-6),5,ifacteaa)
                           s354_tw = fcuhw
                           s354cap = -1.0
                           IF (.not. LFORPUMP(is8bsn-6)) THEN
                              call eaaspill(s354cap,stagelo,s354_tw,5,
     $                        ifacteaa)
                           ELSE
                              call eaaspill(s354cap,stagelo,s354_tw,9,
     $                        ifacteaa)
                           ENDIF
                        ENDIF
                        miamic_cap_upper_grav = total_flow_upper_reach
                     ENDIF
                  ENDIF
               ENDIF
               IF (s8_new_hw .GT. s8_tailw) THEN
                  ireach = 8
                  call qmax_one_reach(s8_new_hw,s8_tailw ,3,1,ireach,1,
     $            flow_lower_reach_max,s8_cap_new ,
     $            miamic_cap_lower_grav,s8_cap,chw2,ctw2,ifacteaa)
c              print *,s8_new_hw,s8_tailw,flow_lower_reach_max
c    $          ,s8_cap_new,miamic_cap_lower_grav,s8_cap,' qmax one rch'
                  s8_new_tw = chw2
                  s8_spill_hw = ctw2
               ELSE
                  flow_lower_reach_max = 0.0
                  s8_cap_new = 0.0
                  miamic_cap_lower_grav = -1.0
c              print *,s8_new_tw,s8_spill_hw,' s8new pump,hw lt wca3a'
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

```
                  IF (s8_new_tw .GT. s8_spill_hw) THEN
                     call eaa_prof_lu_table(miamic_cap_lower_grav ,
     $               s8_new_tw,s8_spill_hw,8,ifacteaa)
                  ELSE
                     miamic_cap_lower_grav = 0.0
                  ENDIF
c              print *,miamic_cap_lower_grav,s8_new_tw,s8_spill_hw,' D'
                  s8_cap = -1.0
                  call eaaspill(s8_cap,s8_spill_hw,s8_tailw, 1,ifacteaa)

               ENDIF
               miamic_cap_lower_grav = miamic_cap_lower_grav - amin1
     $         (miamic_cap_lower_grav,total_outflow_from_roten +
     $         total_outf_from_holy_g200a)
               flow_lower_reach_max = miamic_cap_lower_grav
            ELSE
               total_flow_upper_reach = total_flow_s8new +
     $         dmnd_miami_basin + add_conv_red_miami +
     $         ws_to_sta34_frm_lok + ws_frm_lok_to_semcyp +
     $         ws_frm_lok_to_holy
               IF (total_flow_upper_reach .GT. miamic_cap_upper_pump)
     $         THEN
                  fcuhw = -1.0
                  ireach = 5
                  IF ( flow_s8new_to_sta .GT. 0.0) THEN
                     s8_new_hw = pmp_int_new(is8bsn-6)
                  ELSE
                     s8_new_hw = amax1(s8_new_hw,pmp_int_new(is8bsn-6))
                  ENDIF
                  call eaa_prof_lu_table(total_flow_upper_reach,fcuhw ,
     $            s8_new_hw,ireach,ifacteaa)
                  fcuhw = amin1(fcuhw,rmaxstg(is8bsn-6))
                  s354_tw = fcuhw
                  miamic_cap_upper_pump = -1.0
                  IF (s354_tw .GT. s8_new_hw) THEN
                     call eaa_prof_lu_table(miamic_cap_upper_pump,
     $               s354_tw, s8_new_hw,ireach,ifacteaa)
                  ELSE
                     miamic_cap_upper_pump = 0.0
                  ENDIF
                  s354cap = -1.0
                  call eaaspill(s354cap,stagelo,s354_tw,5,ifacteaa)
               ELSE
                  IF (total_flow_upper_reach .GT. 0.0) THEN
                     IF (flow_s8new_to_sta .GT. 0.0) THEN
                        fcutw = -1.0
                        call eaa_prof_lu_table(total_flow_upper_reach,
     $                  s354_tw ,fcutw,5,ifacteaa)
                        s8_new_hw = amax1(fcutw,pmp_int_new(is8bsn-6))
                        IF (fcutw .LT. pmp_int_new(is8bsn-6)) THEN
                           fcuhw = -1.0
                           call eaa_prof_lu_table
     $                     (total_flow_upper_reach ,fcuhw,pmp_int_new
     $                     (is8bsn-6),5,ifacteaa)
                           s354_tw = fcuhw
                           s354cap = -1.0
                           call eaaspill(s354cap,stagelo,s354_tw,5,
```

```
     $                    ifacteaa)
                       ENDIF
                       miamic_cap_upper_pump = total_flow_upper_reach
                    ENDIF
                 ENDIF
              ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
              istype = 1
              istr_eaa = 3
              ireach = 8
              chw = s8_new_hw
              ctw = pmp_int(is8bsn-6)
              delta0 = (chw-ctw)/25.
              call us_flow(istype,istr_eaa,ireach,1,chw,ctw,s8_new_hw ,
     $        delta0,1.0,100,miamic_cap_lower_pump,s8_cap_new ,
     $        s8_new_tw,ifacteaa)
              s8_spill_hw = pmp_int(is8bsn-6)
              miamic_cap_lower_pump = miamic_cap_lower_pump - amin1
     $        (miamic_cap_lower_pump,total_outflow_from_roten +
     $        total_outf_from_holy_g200a)
              flow_lower_reach_max = amin1(miamic_cap_lower_pump ,
     $        s8_cap_new)
              s8_cap = s8max
           ENDIF
c
           IF (.not. LFORPUMP(is8bsn-6)) THEN
              IF (stagelo .LE. pmp_int_new(is8bsn-6)) THEN
                 add_flow_through = amin1(dmnd_lec,flow_through_unadj)
              ELSE
                 add_flow_through = 0.0
              ENDIF
           ELSE
             add_flow_through = 0.0
           ENDIF
c
           IF (flow_to_wca_prior .EQ. 'LEC') THEN
              IF (stagelo .LE. pmp_int_new(is8bsn-6)) THEN
                 IF (ntarg .GT. 0) THEN
                    flow_through = flow_thru_to_glades + amax1
     $              (flow_to_lec - frac_lec(is8bsn-6)*runoff,0.0)
     $              * frac_del_frm_wca(iwca3a)
                    ws_to_lec = amax1(flow_to_lec-frac_lec(is8bsn-6)
     $              *runoff ,0.0) * frac_del_frm_wca(iwca3a)
                    excess_flow = total_inflow-rmax_inflow+flow_to_lec
     $              -amax1(flow_to_lec-frac_lec(is8bsn-6)
     $              ,0.0)*(1.0-frac_del_frm_wca(iwca3a))
                 ELSE
                    flow_through = flow_thru_to_glades + amax1
     $              (flow_to_lec - runoff,0.0)*frac_del_frm_wca(iwca3a)
                    ws_to_lec = amax1(flow_to_lec - runoff,0.0)
     $              * frac_del_frm_wca(iwca3a)
                    excess_flow = total_inflow-rmax_inflow+flow_to_lec
     $              -amax1(flow_to_lec - runoff,0.0)*(1.0
     $              - frac_del_frm_wca(iwca3a))
                 ENDIF
              ELSE
                 ws_to_lec = amin1(flow_to_lec , flow_through)
```

```
     $            * frac_del_frm_wca(iwca3a)
                  excess_flow = total_inflow-rmax_inflow+flow_to_lec
     $            - amax1(flow_to_lec - runoff,0.0)*(1.0
     $            - frac_del_frm_wca(iwca3a))
               ENDIF
            ELSE IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $      dmnd_lec .AND. total_inflow_prescap.LE.dmnd_lec) THEN
               flow_through = total_inflow + flow_thru_prescap +
     $         add_flow_through * frac_del_frm_wca(iwca3a)
               excess_flow= total_inflow-rmax_inflow
     $         +total_inflow_prescap + add_flow_through
     $         * frac_del_frm_wca(iwca3a)
               ws_to_lec = flow_thru_prescap + add_flow_through
     $         * frac_del_frm_wca(iwca3a)
            ELSE
               excess_flow= total_inflow - rmax_inflow +
     $         add_flow_through * frac_del_frm_wca(iwca3a)
               flow_through = flow_through + add_flow_through
     $         * frac_del_frm_wca(iwca3a)
               ws_to_lec = add_flow_through*frac_del_frm_wca(iwca3a)
            ENDIF
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (excess_flow .GT. flow_lower_reach_max) THEN
c
               iagflo(iag(is8bsn),1) = iagflo(iag(is8bsn),1) + nint
     $         (amax1(excess_flow - flow_through-flow_lower_reach_max ,
     $         0.0))
               flow_through = flow_through - (excess_flow -
     $         flow_lower_reach_max)
               flow_through = amax1(flow_through,0.0)
               ws_to_lec = ws_to_lec-amin1(excess_flow
     $         -flow_lower_reach_max ,ws_to_lec)
            ENDIF
            IF (opt_pipeline_ws(is8bsn-6) .EQ. 'PIPEL  ') THEN
               ws_pipeline_3a = ws_pipeline_3a + amin1(excess_flow ,
     $         flow_lower_reach_max,dmnd_lec)
            ENDIF
c
            kflo(ifls8) = amin1(flow_lower_reach_max,excess_flow) + kflo
     $      (ifls8)
            IF (igravity_s8 .EQ. 1) THEN
               s8capacity = s8_cap
            ELSE
               s8capacity = s8max
            ENDIF
            IF (kflo(ifls8) .GT. s8capacity) THEN
               flow_above_capac = kflo(ifls8) - s8capacity
               kflo(ifls8) = s8capacity
            ELSE
               flow_above_capac = 0.0
            ENDIF
            runoff_to_lec_s8 = amin1(excess_flow,dmnd_lec ,
     $      flow_lower_reach_max) - flow_through - flow_above_capac

            runoff_to_lec_s8 = amax1(runoff_to_lec_s8,0.0)
            emerg_bypass_sta34 = amin1(total_inflow - rmax_inflow ,
```

```
     $      flow_lower_reach_max) - amax1(flow_above_capac -
     $      flow_through,0.0)
            emerg_bypass_sta34 = amax1(emerg_bypass_sta34,0.0)
            emerg_bypass_s8 = emerg_bypass_sta34
            iagflo(iag(is8bsn),1) = iagflo(iag(is8bsn),1) + nint(amax1
     $      (flow_above_capac - flow_through,0.0))
            flow_through = flow_through - amin1(flow_through ,
     $      flow_above_capac)
            kflo(ifls3) = kflo(ifls3) + flow_through
            kfloref(is8bsn-6) = kfloref(is8bsn-6) + flow_through
            wss354 = flow_through + wss354
            if (dmnd_lec .gt. 0.00001) then
              rlok_flow_to_enp_s354 = (amin1(flow_through,ws_to_lec)
     $        /dmnd_lec)*frf
            else
              rlok_flow_to_enp_s354 = 0.0
            endif
            ws_to_lec_s354 = ws_to_lec_s354 + amax1(amin1(ws_to_lec ,
     $      flow_through) - rlok_flow_to_enp_s354,0.0)
            IF (ssm_env .EQ. 'TRUE ') THEN
               dellec = dellec + flow_through * 1.9835
            ELSE
               IF (wcaenv(iwca3a) .EQ. 'FALSE' .OR. ntarg .EQ. 0) THEN
                  dellec = dellec + amin1(flow_through,dmnd_lec)*1.9835
               ENDIF
            ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            flow_from_eaa_lok_s8 = amin1(flow_lower_reach_max ,
     $      excess_flow) - flow_above_capac
            flow_from_eaa_lok_s8 = amax1(flow_from_eaa_lok_s8,0.0)
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               flow_to_import = flow_to_import + flow_thru_to_glades
               flow_to_import_miami = flow_thru_to_glades
            ELSE IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $      dmnd_lec .AND. total_inflow_prescap_unadj.LE.dmnd_lec) THEN
               flow_to_import = flow_to_import + rmax_to_glades
               flow_to_import_miami = rmax_to_glades
            ELSE
               flow_to_import = flow_to_import + amax1(flow_through -
     $         ws_to_lec,0.0)
               flow_to_import_miami = amax1(flow_through - ws_to_lec ,
     $         0.0)
            ENDIF
         ELSE
c
c   proposed s8(s8new) is not pumping into STA: runoff from miami canal
c   basin not  backpumped into LOK is less than LEC demands,thus runoff
c   bypasses STA to meet the LEC demands
c
            flow_s8new_to_sta = 0.0
            IF ( eaa_conv_opt_s8 .EQ. 'PUMP') THEN
               eaa_convey_max = eaa_convey(2,1)
            ELSE
               eaa_convey_max = eaa_convey(1,1)
            ENDIF
            flow_lower_reach_max = amax1(eaa_convey_max -
     $      total_outflow_from_roten -total_outf_from_holy_g200a,0.0)
```

```
            IF (igravity_s8 .EQ. 1) THEN
               s8capacity = s8_cap
            ELSE
               s8capacity = s8max
            ENDIF
            avail_cap_for_sta34_miami = total_inflow
            IF(runoff .GT. flow_lower_reach_max) THEN
               KFLO(IFLS8) = flow_lower_reach_max + kflo(ifls8)
               IF (kflo(ifls8) .GT. s8capacity) THEN
                  flow_above_capac = kflo(ifls8) - s8capacity
                  kflo(ifls8) = s8capacity
               ELSE
                  flow_above_capac = 0.0
               ENDIF
               IAGFLO(IAG(IS8BSN),1) = NINT(runoff-flow_lower_reach_max)
     $         + NINT(flow_above_capac) + IAGFLO(IAG(IS8BSN),1)
               flow_from_eaa_lok_s8 = amax1(flow_lower_reach_max -
     $         flow_above_capac,0.0)
            ELSE
               KFLO(IFLS8) = runoff + kflo(ifls8)
               IF (kflo(ifls8) .GT. s8capacity) THEN
                  flow_above_capac = kflo(ifls8) - s8capacity
                  kflo(ifls8) = s8capacity
               ELSE
                  flow_above_capac = 0.0
               ENDIF
               IAGFLO(IAG(IS8BSN),1) = NINT(flow_above_capac) + IAGFLO
     $         (IAG(IS8BSN),1)
               flow_from_eaa_lok_s8 = amax1(runoff - flow_above_capac,
     $         0.0)
            ENDIF
            runoff_to_lec_s8 = flow_from_eaa_lok_s8
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c        write(*,246) total_inflow,runoff,flow_through,kflo(ifls3)
c    $,   kflo(ifls8),amax1(agp(is8bsn-6)/86400.,0.0),dmnd_lec
c    $,   excess_flow,iagflo(iag(is8bsn),1),month,iday,ifyr+iyear-1
c    $,   qinflw(indxsta(is8bsn-6),ncount(indxsta(is8bsn-6)))
c    $,   rmax_inflow,stagelo,pmp_int_new(is8bsn-6)
c246     format(8f7.0,i7,2x,2i3,i5,2f7.0,2f7.2)
c
c    runoff form s-7 basin into sta3 and sta4
c
      ELSE IF (istopt .EQ. 2) THEN
         ireach = 6
         istype = 1
         ctw = pmp_int_new(is7bsn-6)
         if (.not. LFORPUMP(is7bsn-6)) THEN
            istr_eaa = 6
            chw = stagelo
            delta0 = (chw - ctw) / 25.0
            call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo,delta0
     $      ,1.0,100,qcanal,s351cap_temp,s351_tw_temp ,ifacteaa)

            IF (s351_tw_temp .GT. rmaxstg(is7bsn-6)) THEN
               s351_tw_temp = rmaxstg(is7bsn-6)
               s351cap_temp = -1.0
```

```
               call eaaspill(s351cap_temp,stagelo,rmaxstg(is7bsn-6), 6,
     $         ifacteaa)
               qcanal = -1.0
               IF (s351_tw_temp .GT. ctw) THEN
                  call eaa_prof_lu_table(qcanal,s351_tw_temp,ctw,ireach,
     $            ifacteaa)
               ELSE
                  qcanal = 0.0
               ENDIF
            ENDIF
         ELSE
            istr_eaa = 10
            chw = amin1(rmaxstg(is7bsn-6),stagelo+rmax_lift(is7bsn-6))
            chw_max = chw
            delta0 = (chw - ctw) / 25.0
            call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo,delta0
     $      ,1.0,100,qcanal,s351cap_temp,s351_tw_temp ,ifacteaa)
            IF (s351_tw_temp .GT. chw_max) THEN
               s351_tw_temp = chw_max
               IF (s351_tw_temp .GT. ctw) THEN
                  qcanal = -1.0
                  call eaa_prof_lu_table(qcanal,s351_tw_temp,ctw,ireach,
     $            ifacteaa)
               ELSE
                  qcanal = 0.0
               ENDIF
            ENDIF
         ENDIF
         fllks6_tmp = -1.0
         call eaa_prof_lu_table(fllks6_tmp,s351_tw_temp,pmp_int
     $   (is6bsn-6) ,3,ifacteaa)
         grf2amx_to_sta = amin1(qcanal,s351cap_temp-fllks6_tmp)
         grf2amx_to_sta = amax1(grf2amx_to_sta,0.0)
         grf2amx_prescap_to_sta = grf2amx_to_sta /amax1(rcpfacteaa_canal
     $   (is7bsn-6),1.0)
         excnnr = 0.0
         IF (iter_asr .LE. 1) THEN
            ncount(indxsta(is7bsn-6)) = ncount(indxsta(is7bsn-6)) + 1
         ENDIF
c
         ntarg = 0
         DO itarg = 1,no_targ_loc(is7bsn-6)
            IF (lwslo(itarg_area(itarg,is7bsn-6))) THEN
               ntarg = ntarg + 1
            ENDIF
         END DO
         ntarg_hill = 0
         DO itarg = 1,no_targ_loc(is6bsn-6)
            IF (lwslo(itarg_area(itarg,is6bsn-6))) THEN
               ntarg_hill = ntarg_hill + 1
            ENDIF
         END DO
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
         avail_cap_for_sta34_nnr = amin1(volwtr(indxsta(is7bsn-6)) ,
     $   g370cap*pctwma(indxsta(is7bsn-6)))
         IF (ifacteaa .EQ. 1) THEN
```

```
            flow_thru_prescap = amax1(grf2amx_prescap - kfloref
     $      (is7bsn-6) ,0.0)
            flow_thru_prescap = amin1(flow_thru_prescap,flow_through)
            total_inflow_prescap = runoff + flow_thru_prescap
         ELSE
            total_inflow_prescap = total_inflow
         ENDIF
         total_inflow_prescap_unadj = total_inflow_prescap
         IF (ntarg .gt. 0) THEN
            if(ifacteaa .EQ. 1) flow_thru_prescap_to_sta =
     $      flow_thru_prescap + amax1(grf2amx_prescap_to_sta -
     $      grf2amx_prescap,0.0)
            capac_incrse = amax1(grf2amx_to_sta - grf2amx,0.0)
c        capac_incrse = amin1(capac_incrse,avail_cap_for_sta34_nnr
c    $                      - (runoff+flow_through))
            flow_through_to_sta = flow_through + capac_incrse
         ELSE
            if(ifacteaa .EQ. 1) flow_thru_prescap_to_sta =
     $      flow_thru_prescap
            flow_through_to_sta = flow_through
         ENDIF
         total_flow_env = runoff + flow_through_to_sta
         IF (iconvey_canal_no(is150bws-6) .ne. iconvey_canal_no
     $    (is8bsn-6)) then
            indx_wca = iwca3a
            rmin_stg_wca0 = rmin_stg_wca3a
            ieaa_bsn0 = is8bsn-6
         ELSE
            indx_wca = iwca2a
            rmin_stg_wca0 = rmin_stg_wca2a
            ieaa_bsn0 = is7bsn-6
         ENDIF
c        print *,indx_wca,iwca3a,iwca2a,rmin_stg_wca0,
c    $   total_inflow_prescap,total_inflow,LWSLO_floor(indx_wca),
c    $   LWSLECSTA(ieaa_bsn0),rmnstg(indx_wca,iconvey_canal_no_wca
c    $   (indx_wca)),dmnd_lec,'  POPOPO'
         IF ((total_inflow_prescap .GT. dmnd_lec .OR. (ifacteaa .EQ. 1
     $   .AND. total_inflow .GT. dmnd_lec)).OR.((((LWSLO_floor(indx_wca)
     $   .AND. LWSLECSTA(ieaa_bsn0).AND.floor_grid_cell_opt.eq.'MARSH')
     $   .OR. rmin_stg_wca0 .gt.rmnstg(indx_wca,iconvey_canal_no_wca
     $   (indx_wca))+0.001)).AND. iopt_ws_lec_to_sta_wca_abv_flr .eq. 1)
     $   .OR. flow_to_wca_prior .EQ.'LEC') THEN
c        IF (total_inflow_prescap .GT. dmnd_lec .OR. (ifacteaa.EQ. 1.and
c    $   . total_inflow .GT. dmnd_lec) .OR. flow_to_wca_prior .EQ. 'LEC'
c    $   ) THEN
            s351_tw = s351_tw_temp
            s351cap = s351cap_temp
            s351mx = s351cap
            fllks6 = fllks6_tmp
            nnrc_cap_upper_grav = grf2amx_to_sta
            nnrc_cap_upper_pump = grf2amx_to_sta
            oper_capac = cnlcap(innr) - (1.0-pct_des(innr,iseas))
     $      *cnlcap(innr)/rcpfacteaa(is7bsn-6)
            IF (.not. LFORPUMP(is7bsn-6)) THEN
               IF (stagelo .LE. pmp_int_new(is7bsn-6)) THEN
                  total_inflow = runoff
                  flow_through = 0.0
```

```
                  total_inflow_prescap = runoff + flow_through
                  flow_thru_prescap = flow_through
               ENDIF
            ENDIF
            IF (.NOT. DRYSEAS .AND. nint(runoff) .GT. 0) THEN
               rmax_to_glades = amin1(grf2amx_to_sta,oper_capac) -
     $         kfloref(is7bsn-6)
            ELSE
               rmax_to_glades = amin1(grf2amx_to_sta-kfloref(is7bsn-6) ,
     $         oper_capac)
            ENDIF
            rmax_to_glades = amax1(rmax_to_glades,0.0)
            rmax_to_glades = amin1(rmax_to_glades ,
     $      avail_cap_for_sta34_nnr)
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               rem_ws_needs_3a = amax1(dmnd_lec_sa3 - kflo(ifls8) -
     $         ws_pipeline_3a,0.0)
               total_ds_needs = dmnd_lec
               IF (stagelo .GT. pmp_int_new(is7bsn-6)) THEN
                  IF (ntarg .GT. 0 .AND. nnrctwca3a .EQ. 'TRUE') THEN
                     avail_flow_to_lec = frac_lec(is7bsn-6)*total_inflow
                  ELSE
                     avail_flow_to_lec = total_inflow
                  ENDIF
                  flow_to_lec = amin1(avail_flow_to_lec,total_ds_needs)
                  rem_to_glades = total_flow_env - flow_to_lec
               ELSE
                  IF (ntarg .GT. 0 .AND. nnrctwca3a .EQ. 'TRUE') THEN
                     avail_flow_to_lec = flow_through_unadj + frac_lec
     $               (is7bsn-6)*total_inflow
                     rem_to_glades = total_flow_env - amin1(frac_lec
     $               (is7bsn-6)*total_inflow,total_ds_needs)
                  ELSE
                     avail_flow_to_lec = flow_through_unadj +
     $               total_inflow
                     rem_to_glades = total_flow_env - amin1
     $               (total_inflow ,total_ds_needs)
                  ENDIF
               ENDIF
               rem_to_glades = amin1(rem_to_glades,
     $         avail_cap_for_sta34_nnr)
               flow_thru_to_glades_bfop = amax1(rem_to_glades-runoff,
     $         0.0)
               flow_thru_to_glades = amin1(flow_thru_to_glades_bfop ,
     $         rmax_to_glades)
               total_inflow = amin1(runoff,rem_to_glades) +
     $         flow_thru_to_glades
               flow_through = flow_through_to_sta
     $         -(flow_thru_to_glades_bfop - flow_thru_to_glades)
            ELSE
               IF (nasr_to_lok(1) .EQ. 0) THEN
                  call bpts(iday,month,10.0,1,n_fc_zones+4,1,
     $            stgref_asr_rec)
                  IF (stagelo .LT. stgref_asr_rec .AND.(climate_index
     $            (month) .LT. 1.5 .OR. ((itrib_hydro_cond_opt .EQ. 1
```

```
     $            .AND. trib_rf_et .LT. -1.0 .AND.
     $            s65e_runff_wkly .LT. 1500.) .OR.
     $            itrib_hydro_cond_opt .EQ. 0)) .AND. DRYSEAS) THEN
                     frac_env_deliv = 0.00
                  ELSE
                     frac_env_deliv = 1.00
                  ENDIF
               ELSE
                  frac_env_deliv = 1.00
               ENDIF
               frac_env_deliv = amin1(frac_env_deliv,frac_env_met)
               IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $         dmnd_lec .AND. total_inflow_prescap_unadj .LE.dmnd_lec)
     $         THEN
                  flow_thru_glades = amax1(flow_through_to_sta
     $            -flow_thru_prescap,0.0)
                  rmax_to_glades = amin1(rmax_to_glades,
     $            flow_thru_glades) * frac_rem_capac(is7bsn-6) *
     $            frac_env_deliv
                  total_inflow = amin1(rmax_to_glades ,
     $            avail_cap_for_sta34_nnr)
               ELSE
                  flow_through = amin1(flow_through_to_sta,
     $            rmax_to_glades) * frac_rem_capac(is7bsn-6) *
     $            frac_env_deliv
                  total_inflow = runoff + flow_through
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                  IF (runoff .LE. avail_cap_for_sta34_nnr) THEN
                     flow_through = flow_through - (total_inflow - amin1
     $               (total_inflow,avail_cap_for_sta34_nnr))
                     total_inflow = amin1(total_inflow ,
     $               avail_cap_for_sta34_nnr)
                  ELSE
                     total_inflow = runoff
                     flow_through = 0.0
                  ENDIF
               ENDIF
            ENDIF
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            qinflw(indxsta(is7bsn-6),ncount(indxsta(is7bsn-6)))= amin1
     $      (volwtr (indxsta(is7bsn-6)),total_inflow,g370cap*pctwma
     $      (indxsta(is7bsn -6)))
c
c    compute excess runoff from s-7 basin sta34 could not take
c
            rem_runoff_over_des_news7 = amax1(total_inflow - g370cap *
     $      pctwma(indxsta(is7bsn-6)),0.0)
            rem_runoff_over_stor_sta34 = amax1(total_inflow - volwtr
     $      (indxsta(is7bsn-6)),0.0) + rem_runoff_over_stor_sta34
c
c
c   compute remaining available storage in sta34
c
            IF (ntarg_hill.GT. 0 .AND. (iasr_opt_eaa(is7bsn-6) .EQ. 1
     $      .OR. ires_opt_hill .EQ. 1)) THEN
               IF (conv_asr .EQ. 1) THEN
```

```
                  volwtr(indxsta(is7bsn-6)) = volwtr(indxsta(is7bsn-6))
     $            - qinflw(indxsta(is7bsn-6),ncount(indxsta(is7bsn-6)))
               ENDIF
            ELSE
               volwtr(indxsta(is7bsn-6)) = volwtr(indxsta(is7bsn-6)) -
     $         qinflw(indxsta(is7bsn-6),ncount(indxsta(is7bsn-6)))
            ENDIF
c
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               excnnr = total_inflow - qinflw(indxsta(is7bsn-6) ,ncount
     $         (indxsta(is7bsn-6))) + flow_to_lec - amax1(flow_to_lec
     $         - runoff,0.0) * (1.0 - frac_del_frm_wca(iwca2a))
            ELSE
               IF (.not. LFORPUMP(is7bsn-6)) THEN
                  IF (stagelo .LE. pmp_int_new(is7bsn-6)) THEN
                     add_flow_thru = amin1(dmnd_lec,flow_through_unadj)
                  ELSE
                     add_flow_thru = 0.0
                  ENDIF
               ELSE
                  add_flow_thru = 0.0
               ENDIF
               IF (ifacteaa.EQ. 1 .AND. total_inflow_unadj .GT.
     $         dmnd_lec .AND. total_inflow_prescap_unadj.LE.dmnd_lec)
     $         THEN
                  excnnr= total_inflow - qinflw(indxsta(is7bsn-6) ,
     $            ncount(indxsta(is7bsn-6))) + total_inflow_prescap +
     $            add_flow_thru * frac_del_frm_wca(iwca2a)
     $            - amax1(total_inflow_prescap - runoff,0.0)
     $            * (1.0 - frac_del_frm_wca(iwca2a))
               ELSE
                  excnnr= total_inflow - qinflw(indxsta(is7bsn-6) ,
     $            ncount(indxsta(is7bsn-6))) + add_flow_thru
     $            * frac_del_frm_wca(iwca2a)
               ENDIF
            ENDIF
            flow_s7new_to_sta = qinflw(indxsta(is7bsn-6) ,ncount(indxsta
     $      (is7bsn-6)))
            rnnrc_runoff_to_sta = amin1(qinflw(indxsta(is7bsn-6) ,ncount
     $      (indxsta(is7bsn-6))),runoff)
            IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT. dmnd_lec
     $      .AND. total_inflow_prescap_unadj .LE. dmnd_lec)
     $      rnnrc_runoff_to_sta = 0.0
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            flow_lower_reach_max = -1.0
            ireach = 9
            total_flow_s7new = flow_s7new_to_sta + excnnr
            IF (eaa_conv_opt_s7 .NE. 'PUMP' .AND. runoff .LE.
     $      avail_cap_for_sta34_nnr) THEN
c   water supply releases by gravity only thru existing S7
c
c   recompute canal profile for upper reach of nnrc when proposed s7
c   (s7new) is pumping into STA
c
c           print *,total_flow_s7new,nnrc_cap_upper_grav,'  s7new pump'
               total_flow_upper_reach = total_flow_s7new + dmnd_nnr_orig
```

```
               IF (total_flow_upper_reach .GT. nnrc_cap_upper_grav) THEN
                  fcuhw = -1.0
                  ireach = 6
                  IF (flow_s7new_to_sta .GT. 0.0) THEN
                     s7_new_hw = pmp_int_new(is7bsn-6)
                  ELSE
                     s7_new_hw = amax1(s7_new_hw,pmp_int_new(is7bsn-6))
                  ENDIF
                  call eaa_prof_lu_table(total_flow_upper_reach,fcuhw ,
     $            s7_new_hw,ireach,ifacteaa)
                  IF (.not. LFORPUMP(is7bsn-6)) THEN
                     fcuhw = amin1(fcuhw,rmaxstg(is7bsn-6))
                  ELSE
                     fcuhw = amin1(fcuhw,rmaxstg(is7bsn-6),
     $               stagelo+rmax_lift(is7bsn-6))
                  ENDIF
                  s351_tw = fcuhw
c             print *,s351_tw,fcuhw,rmaxstg(is7bsn-6),'  A'
c
c   recompute canal capacity for hillsboro canal when s7new is pumping
c   water into STA3 and 4
c
                  fllks6 = -1.0
                  IF (s351_tw .GT. pmp_int(is6bsn-6)) THEN
                     call eaa_prof_lu_table(fllks6 ,s351_tw,pmp_int
     $               (is6bsn-6),7,ifacteaa)
                  ELSE
                     fllks6 = 0.0
                  ENDIF
c
                  nnrc_cap_upper_grav = -1.0
                  IF (s351_tw.GT.s7_new_hw) THEN
                     call eaa_prof_lu_table(nnrc_cap_upper_grav,s351_tw,
     $               s7_new_hw,ireach,ifacteaa)
                  ELSE
                     nnrc_cap_upper_grav = 0.0
                  ENDIF
                  s351cap = -1.0
                  IF (.not. LFORPUMP(is7bsn-6)) THEN
                     call eaaspill(s351cap,stagelo,s351_tw,6,ifacteaa)
                  ELSE
                     call eaaspill(s351cap,stagelo,s351_tw,10,ifacteaa)
                  ENDIF
c              print *,s351cap,stagelo,s351_tw,nnrc_cap_upper_grav,'  B'
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               ELSE
                  IF (total_flow_upper_reach .GT. 0.0) THEN
                     IF (flow_s7new_to_sta .GT. 0.0) THEN
                        fcutw = -1.0
                        call eaa_prof_lu_table(total_flow_upper_reach,
     $                  s351_tw ,fcutw,6,ifacteaa)
                        s7_new_hw = amax1(fcutw,pmp_int_new(is7bsn-6))
                        IF (fcutw .LT. pmp_int_new(is7bsn-6)) THEN
                           fcuhw = -1.0
                           call eaa_prof_lu_table
     $                     (total_flow_upper_reach,fcuhw ,pmp_int_new
```

```
     $                     (is7bsn-6),6,ifacteaa)
                           s351_tw = fcuhw
                           s351cap = -1.0
                           IF (.not. LFORPUMP(is7bsn-6)) THEN
                              call eaaspill(s351cap,stagelo,s351_tw,6,
     $                        ifacteaa)
                           ELSE
                              call eaaspill(s351cap,stagelo,s351_tw,10,
     $                        ifacteaa)
                           ENDIF
c
c   recompute canal capacity for hillsboro canal when s7new is pumping
c   water into STA3 and 4
c
                           fllks6 = -1.0
                           IF (s351_tw.GT.pmp_int(is6bsn-6)) THEN
                              call eaa_prof_lu_table(fllks6 ,s351_tw,
     $                        pmp_int(is6bsn-6),7,ifacteaa)
                           ELSE
                              fllks6 = 0.0
                           ENDIF
                        ENDIF
                        nnrc_cap_upper_grav = total_flow_upper_reach
c             print *,total_flow_s7new,nnrc_cap_upper_grav,fcutw,s351_tw
c    $          ,'  C'
                     ENDIF
                  ENDIF
               ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               IF (s7_new_hw .GT. s7_tailw) THEN
                  call qmax_one_reach(s7_new_hw,s7_tailw ,4,2,ireach,1,
     $            flow_lower_reach_max,s7_cap_new ,nnrc_cap_lower_grav,
     $            s7_cap,chw2,ctw2,ifacteaa)
                  s7_new_tw = chw2
                  s7_spill_hw = ctw2
               ELSE
                  flow_lower_reach_max = 0.0
                  s7_cap_new = 0.0
                  nnrc_cap_lower_grav = -1.0
                  IF (s7_new_tw.GT.s7_spill_hw) THEN
                     call eaa_prof_lu_table(nnrc_cap_lower_grav,
     $               s7_new_tw ,s7_spill_hw,8,ifacteaa)
                  ELSE
                     nnrc_cap_lower_grav = 0.0
                  ENDIF
                  s7_cap = -1.0
                  call eaaspill(s7_cap,s7_spill_hw,s7_tailw,2,ifacteaa)
               ENDIF
               IF ((iconvey_canal_no(is150bws-6) .NE. iconvey_canal_no
     $         (is8bsn-6) .AND. idn_wca_indx(is150bws-6) .EQ.
     $         idn_wca_indx(is8bsn-6) .AND. iconvey_canal_no(is150bws-6)
     $         .NE. 0) .OR. opt_pipeline_ws(is150bws-6) .EQ. 'PIPEL  ')
     $         THEN
                  IF (s7_new_hw .GT. s150_tailw) THEN
                     call qmax_one_reach(s7_new_hw,s150_tailw ,4,2,
     $               ireach,1,flow_lower_reach_max_s150,s7_cap_new ,
     $               nnrc_cap_lower_grav_s150,s150cap,chw2,ctw2,
```

```
     $               ifacteaa)
                     s7_new_tw = chw2
                     s7_spill_hw = ctw2
                  ELSE
                     flow_lower_reach_max_s150 = 0.0
                     s7_cap_new = 0.0
                     s150cap = 0.0
                     nnrc_cap_lower_grav_s150 = -1.0
                     IF (s7_new_tw .GT. s7_spill_hw) THEN
                        call eaa_prof_lu_table
     $                  (nnrc_cap_lower_grav_s150 ,s7_new_tw,
     $                  s7_spill_hw,8,ifacteaa)
                     ELSE
                        nnrc_cap_lower_grav_s150 = 0.0
                     ENDIF
                  ENDIF
               ENDIF
            ELSE
c
c   existing S7 is pumping water supply releases
c
c
c   recompute canal profile for upper reach of nnrc when s7new is
c   pumping
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               total_flow_upper_reach = total_flow_s7new + dmnd_nnr_orig
               IF (total_flow_upper_reach .GT. nnrc_cap_upper_pump) THEN
                  fcuhw = -1.0
                  ireach = 6
                  IF (flow_s7new_to_sta .GT. 0.0) THEN
                     s7_new_hw = pmp_int_new(is7bsn-6)
                  ELSE
                     s7_new_hw = amax1(pmp_int_new(is7bsn-6),s7_new_hw)
                  ENDIF
                  call eaa_prof_lu_table(total_flow_upper_reach,fcuhw ,
     $            s7_new_hw,ireach,ifacteaa)
                  if (.not. LFORPUMP(is7bsn-6)) THEN
                     fcuhw = amin1(fcuhw,rmaxstg(is7bsn-6))
                  ELSE
                     fcuhw = amin1(fcuhw,rmaxstg(is7bsn-6),
     $               stagelo+rmax_lift(is7bsn-6))
                  ENDIF
                  s351_tw = fcuhw
c
c   recompute canal capacity for hillsboro canal when s7new is pumping
c   water into STA3 and 4
c
                  fllks6 = -1.0
                  IF (s351_tw .GT. pmp_int(is6bsn-6)) THEN
                     call eaa_prof_lu_table(fllks6 ,s351_tw,pmp_int
     $               (is6bsn-6),7,ifacteaa)
                  ELSE
                     fllks6 = 0.0
                  ENDIF
c
                  nnrc_cap_upper_pump = -1.0
```

```
                  IF (s351_tw .GT. s7_new_hw) THEN
                     call eaa_prof_lu_table(nnrc_cap_upper_pump,s351_tw,
     $               s7_new_hw,ireach,ifacteaa)
                  ELSE
                     nnrc_cap_upper_pump = 0.0
                  ENDIF
                  s351cap = -1.0
                  IF (.not. LFORPUMP(is7bsn-6)) THEN
                     call eaaspill(s351cap,stagelo,s351_tw,6,ifacteaa)
                  ELSE
                     call eaaspill(s351cap,stagelo,s351_tw,10,ifacteaa)
                  ENDIF
               ELSE
                  IF (total_flow_upper_reach .GT. 0.0) THEN
                     IF (flow_s7new_to_sta .GT. 0.0) THEN
                        fcutw = -1.0
                        call eaa_prof_lu_table(total_flow_upper_reach,
     $                  s351_tw ,fcutw,6,ifacteaa)
                        s7_new_hw = amax1(fcutw,pmp_int_new(is7bsn-6))
                        IF (fcutw .LT. pmp_int_new(is7bsn-6)) THEN
                           fcuhw = -1.0
                           call eaa_prof_lu_table
     $                     (total_flow_upper_reach,fcuhw ,pmp_int_new
     $                     (is7bsn-6),6,ifacteaa)
                           s351_tw = fcuhw
c
c   recompute canal capacity for hillsboro canal when s7new is pumping
c   water into STA3 and 4
c
                           fllks6 = -1.0
                           IF (s351_tw.GT.pmp_int(is6bsn-6)) THEN
                              call eaa_prof_lu_table(fllks6 ,s351_tw,
     $                        pmp_int(is6bsn-6),7,ifacteaa)
                           ELSE
                              fllks6 = 0.0
                           ENDIF
                           s351cap = -1.0
                           IF (.not. LFORPUMP(is7bsn-6)) THEN
                              call eaaspill(s351cap,stagelo,s351_tw,6,
     $                        ifacteaa)
                           ELSE
                              call eaaspill(s351cap,stagelo,s351_tw,10,
     $                        ifacteaa)
                           ENDIF
                        ENDIF
                        nnrc_cap_upper_pump = total_flow_upper_reach
                     ENDIF
                  ENDIF
               ENDIF
c
c  recompute structure and canal capacities for lower reach of nnrc
c  when s7new is pumping into STA3 and 4
c
               istype = 1
               istr_eaa = 4
               ireach = 9
               chw = s7_new_hw
```

```
               ctw = pmp_int(is7bsn-6)
               delta0 = (chw-ctw)/25.
               call us_flow(istype,istr_eaa,ireach,2,chw,ctw,s7_new_hw ,
     $         delta0,1.0,100,nnrc_cap_lower_pump,s7_cap_new ,s7_new_tw,
     $         ifacteaa)
               s7_spill_hw = pmp_int(is7bsn-6)
               flow_lower_reach_max= amin1(nnrc_cap_lower_pump,
     $         s7_cap_new)
               s7_cap = s7max
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c  compute conveyance for lower reach for S150 outflow if NNRC is
c  only conveyance canal for water supply for SA3
c
               IF ((iconvey_canal_no(is150bws-6) .NE. iconvey_canal_no
     $         (is8bsn-6) .AND. idn_wca_indx(is150bws-6) .EQ.
     $         idn_wca_indx(is8bsn-6) .AND. iconvey_canal_no(is150bws-6)
     $         .NE. 0) .OR. opt_pipeline_ws(is150bws-6) .EQ. 'PIPEL  ')
     $         THEN
                  IF (s7_new_hw .GT. s150_tailw) THEN
                     call qmax_one_reach(s7_new_hw,s150_tailw ,4,2,
     $               ireach,1,flow_lower_reach_max_s150,s7_cap_new ,
     $               nnrc_cap_lower_grav_s150,s150cap,chw2,ctw2,
     $               ifacteaa)
                     s7_new_tw = chw2
                     s7_spill_hw = ctw2
                  ELSE
                     flow_lower_reach_max_s150 = 0.0
                     s150cap = 0.0
                     s7_cap_new = 0.0
                     nnrc_cap_lower_grav_s150 = -1.0
                     IF (s7_new_tw .GT. s7_spill_hw) THEN
                        call eaa_prof_lu_table
     $                  (nnrc_cap_lower_grav_s150 ,s7_new_tw,
     $                  s7_spill_hw,8,ifacteaa)
                     ELSE
                        nnrc_cap_lower_grav_s150 = 0.0
                     ENDIF
                  ENDIF
c
               ENDIF
            ENDIF
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               IF (stagelo .LE. pmp_int_new(is7bsn-6)) THEN
                  IF (ntarg .GT.  0.AND. nnrctwca3a .EQ. 'TRUE') THEN
                     flow_through = flow_thru_to_glades + amax1
     $               (flow_to_lec - frac_lec(is7bsn-6)*runoff,0.0)
     $               * frac_del_frm_wca(iwca2a)
                     ws_to_lec = amax1(flow_to_lec - frac_lec(is7bsn-6)
     $               *runoff ,0.0) * frac_del_frm_wca(iwca2a)
                  ELSE
                     flow_through = flow_thru_to_glades + amax1
     $               (flow_to_lec - runoff,0.0)*frac_del_frm_wca(iwca2a)
                     ws_to_lec = amax1(flow_to_lec - runoff,0.0)
     $               * frac_del_frm_wca(iwca2a)
```

```
                 ENDIF
              ELSE
                 ws_to_lec = amin1(flow_to_lec,flow_through)
     $           * frac_del_frm_wca(iwca2a)
              ENDIF
           ELSE IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $     dmnd_lec .AND. total_inflow_prescap_unadj.LE.dmnd_lec)
     $         THEN
              flow_through = total_inflow + flow_thru_prescap +
     $        add_flow_thru*frac_del_frm_wca(iwca2a)
     $        - amax1(flow_thru_prescap - runoff,0.0)
     $        * (1.0 - frac_del_frm_wca(iwca2a))
              ws_to_lec = flow_thru_prescap + add_flow_thru
     $        * frac_del_frm_wca(iwca2a) - amax1(flow_thru_prescap
     $        - runoff,0.0) * (1.0 - frac_del_frm_wca(iwca2a))
           ELSE
              flow_through = flow_through + add_flow_thru
     $        * frac_del_frm_wca(iwca2a)
              ws_to_lec = add_flow_thru*frac_del_frm_wca(iwca2a)
           ENDIF
           IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
     $     (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ. idn_wca_indx
     $     (is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6) .NE. 'PIPEL  '
     $     ) THEN

              IF (excnnr .GT. flow_lower_reach_max) THEN
                 IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6)
     $           .EQ. 1 .OR. ires_opt_hill .EQ. 1)) THEN
                    IF (conv_asr .EQ. 1) THEN
                       iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) +
     $                 nint(amax1(excnnr - flow_through
     $                 -flow_lower_reach_max,0.0))
                    ENDIF
                 ELSE
                    iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) +
     $              nint(amax1(excnnr - flow_through
     $              -flow_lower_reach_max,0.0))
                 ENDIF
                 flow_through = flow_through - amin1(excnnr
     $           -flow_lower_reach_max ,flow_through)
                 ws_to_lec = ws_to_lec - amin1(excnnr
     $           -flow_lower_reach_max ,ws_to_lec)
              ENDIF
              excnnr_s7 = excnnr
              flow_through_s7 = flow_through
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
           ELSE
              excnnr_s150 = amin1(excnnr,dmnd_lec_sa3_s150)
              excnnr_s7 = excnnr - excnnr_s150
              flow_through_s7 = amax1(flow_through -
     $        dmnd_lec_sa3_s150 ,0.0)
              IF (excnnr_s7 .GT. flow_lower_reach_max) THEN
                 IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6)
     $           .EQ. 1 .OR. ires_opt_hill .EQ. 1)) THEN
                    IF (conv_asr .EQ. 1) THEN
                       iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) +
     $                 nint(amax1(excnnr_s7 - flow_through_s7
```

```
$                  -flow_lower_reach_max,0.0))
                ENDIF
             ELSE
                iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) +
$               nint(amax1(excnnr_s7 - flow_through_s7
$               -flow_lower_reach_max ,0.0))
             ENDIF
             flow_through = flow_through - amin1(excnnr_s7
$            -flow_lower_reach_max,flow_through)
             ws_to_lec = ws_to_lec - amin1(excnnr_s7
$            -flow_lower_reach_max ,ws_to_lec)
          ENDIF
          IF (opt_pipeline_ws(is7bsn-6) .EQ. 'PIPEL  ') THEN
             ws_pipeline_2a = ws_pipeline_2a + amin1(excnnr_s7 ,
$            dmnd_lec_sa2_s7,flow_lower_reach_max)
          ENDIF
          flow_lower_reach_max_s150 = amax1
$         (flow_lower_reach_max_s150 - amin1(excnnr_s7,
$         flow_lower_reach_max),0.0)
          IF (excnnr_s150 .GT. flow_lower_reach_max_s150) THEN
             flow_through = flow_through - amin1(excnnr_s150,
$            flow_lower_reach_max_s150,flow_through)
             ws_to_lec = ws_to_lec - amin1(excnnr_s150
$            -flow_lower_reach_max_s150,ws_to_lec)
          ENDIF
          IF (opt_pipeline_ws(is150bws-6) .EQ. 'PIPEL  ') THEN
             ws_pipeline_3a_s150 = ws_pipeline_3a_s150 + amin1
$            (excnnr_s150,flow_lower_reach_max_s150 ,
$            dmnd_lec_sa3_s150)
          ENDIF
       ENDIF
       IF (flow_to_wca_prior .EQ. 'LEC') THEN
          s150cap = -1.0
          call eaaspill(s150cap,s7_spill_hw,s150_tailw,8,ifacteaa)
          rem_cap_s150 = s150cap - kflo(ifls150)
          IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
$         (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ.
$         idn_wca_indx(is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6)
$         .NE. 'PIPEL  ') THEN

             rmax_lower_reach = amin1(flow_lower_reach_max,excnnr)
             IF (total_ds_needs .GT. 0.0) THEN
                ratio_wsdel_to_needs=amin1(rmax_lower_reach
$               /total_ds_needs ,1.0)
             ELSE
                ratio_wsdel_to_needs = 1.0
             ENDIF
             rmax_flow_s150 = amin1(ratio_wsdel_to_needs
$            *rem_ws_needs_3a ,rem_cap_s150)
             IF (ntarg_hill.GT. 0 .AND. (iasr_opt_eaa(is7bsn-6)
$            .EQ. 1 .OR. ires_opt_hill .EQ. 1)) THEN
                flow_s150_tmp = rmax_flow_s150 + kflo(ifls150)
                IF (conv_asr .EQ. 1) THEN
                   kflo(ifls150) = flow_s150_tmp
                ENDIF
             ELSE
                kflo(ifls150) = rmax_flow_s150 + kflo(ifls150)
```

```
                  ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                  kflo(ifls7) = rmax_lower_reach - rmax_flow_s150 +
     $            dischg_to_s7_frm_sta34
               ELSE
                  kflo(ifls7) = amin1(flow_lower_reach_max ,amax1
     $            (excnnr - dmnd_lec_sa3_s150,0.0)) +
     $            dischg_to_s7_frm_sta34
                  rem_canal_cap_s150 = amax1(flow_lower_reach_max_s150
     $            -(kflo(ifls7)-dischg_to_s7_frm_sta34),0.0)
                  rem_cap_s150 = s150cap - kflo(ifls150)
                  rmax_flow_s150 = amin1(rem_canal_cap_s150,
     $            rem_cap_s150 ,dmnd_lec_sa3_s150)
                  flow_above_capac_s150 = amin1(rem_canal_cap_s150 ,
     $            dmnd_lec_sa3_s150) - rem_cap_s150
                  flow_above_capac_s150 = amax1(flow_above_capac_s150,
     $            0.0)
                  iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) + nint
     $            (amax1(flow_above_capac_s150-flow_through,0.0))
                  flow_through = flow_through - amin1(flow_through ,
     $            flow_above_capac_s150)
                  IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6)
     $            .EQ. 1.OR. ires_opt_hill .EQ. 1)) THEN
                     flow_s150_tmp = rmax_flow_s150 + kflo(ifls150)
                     IF (conv_asr .EQ. 1) THEN
                        kflo(ifls150) = flow_s150_tmp
                     ENDIF
                  ELSE
                     kflo(ifls150) = rmax_flow_s150 + kflo(ifls150)
                  ENDIF
               ENDIF
               IF (igravity_s7 .EQ. 1) THEN
                  s7capacity = s7_cap
               ELSE
                  s7capacity = s7max
               ENDIF
               IF (kflo(ifls7) .GT. s7capacity) THEN
                  flow_above_capac = kflo(ifls7) - s7capacity
                  kflo(ifls7) = s7capacity
               ELSE
                  flow_above_capac = 0.0
               ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               runoff_to_lec_s7 = amin1(excnnr_s7,dmnd_lec_sa2_s7,
     $         flow_lower_reach_max) - flow_through_s7 -
     $         flow_above_capac
               runoff_to_lec_s7 = amax1(runoff_to_lec_s7,0.0)
               iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) + nint
     $         (amax1(flow_above_capac-flow_through,0.0))
               flow_through = flow_through - amin1(flow_through ,
     $         flow_above_capac)
               IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6) .EQ.
     $         1 .OR. ires_opt_hill .EQ. 1)) THEN
                  flow_s2_tmp = kflo(ifls2) + flow_through
                  IF (conv_asr .EQ. 1) THEN
                     kflo(ifls2) = flow_s2_tmp
                     wss351 = wss351 + flow_through
```

```
                  kfloref(is7bsn-6) = kfloref(is7bsn-6) +
     $            flow_through
               ENDIF
            ELSE
               kflo(ifls2) = kflo(ifls2) + flow_through
               flow_s2_tmp = kflo(ifls2)
               kfloref(is7bsn-6) = kfloref(is7bsn-6) + flow_through
               wss351 = wss351 + flow_through
            ENDIF
            IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6) .EQ.
     $      1 .OR. ires_opt_hill .EQ. 1)) THEN
               IF (conv_asr .EQ. 1) THEN
                  IF (ssm_env .EQ. 'TRUE ') THEN
                     dellec = dellec + wss351 * 1.9835
                  ELSE
                     dellec = dellec + amin1(wss351,total_ds_needs) *
     $               1.9835
                  ENDIF
               ENDIF
            ELSE
               IF (ssm_env .EQ. 'TRUE ') THEN
                  dellec = dellec + wss351 * 1.9835
               ELSE
                  dellec = dellec + amin1(wss351,total_ds_needs) *
     $            1.9835
               ENDIF
            ENDIF
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
     $      (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ.
     $      idn_wca_indx(is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6)
     $      .NE. 'PIPEL  ') THEN

               flow_from_eaa_lok_s7 = rmax_lower_reach -
     $         rmax_flow_s150 - flow_above_capac
            ELSE
               flow_from_eaa_lok_s7 = amin1(flow_lower_reach_max ,
     $         amax1(excnnr-dmnd_lec_sa3_s150,0.0))
            ENDIF
            flow_from_eaa_lok_s7 = amax1(flow_from_eaa_lok_s7,0.0)
         ELSE
            IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
     $      (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ.
     $      idn_wca_indx(is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6)
     $      .NE. 'PIPEL  ') THEN
               kflo(ifls7) = amin1(flow_lower_reach_max,excnnr) +
     $         dischg_to_s7_frm_sta34
            ELSE
               kflo(ifls150) = amin1(excnnr,
     $         flow_lower_reach_max_s150 ,dmnd_lec_sa3_s150)
               excnnr = excnnr - kflo(ifls150)
               flow_lower_reach_max = flow_lower_reach_max - kflo
     $         (ifls150)
               flow_lower_reach_max = amax1(flow_lower_reach_max,0.0)
               kflo(ifls7) = amin1(flow_lower_reach_max,excnnr) +
     $         dischg_to_s7_frm_sta34
```

```
                ENDIF
                IF (igravity_s7 .EQ. 1) THEN
                   s7capacity = s7_cap
                ELSE
                   s7capacity = s7max
                ENDIF
                IF (kflo(ifls7) .GT. s7capacity) THEN
                   flow_above_capac = kflo(ifls7) - s7capacity
                   kflo(ifls7) = s7capacity
                ELSE
                   flow_above_capac = 0.0
                ENDIF
                iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) + nint
     $          (amax1(flow_above_capac-flow_through,0.0))
                flow_through = flow_through - amin1(flow_through ,
     $          flow_above_capac)
                IF (ntarg_hill .GT. 0 .AND. (iasr_opt_eaa(is7bsn-6).EQ.
     $          1 .OR. ires_opt_hill .EQ. 1)) THEN
                   flow_s2_tmp = kflo(ifls2) + flow_through
                   IF (conv_asr .EQ. 1) THEN
                      kflo(ifls2) = flow_s2_tmp
                      wss351 = wss351 + flow_through
                      kfloref(is7bsn-6) = kfloref(is7bsn-6) +
     $                flow_through
                   ENDIF
                ELSE
                   kflo(ifls2) = kflo(ifls2) + flow_through
                   kfloref(is7bsn-6) = kfloref(is7bsn-6) + flow_through
                   flow_s2_tmp = kflo(ifls2)
                   wss351 = wss351 + flow_through
                ENDIF
                IF (ntarg_hill.GT. 0 .AND. (iasr_opt_eaa(is7bsn-6) .EQ.
     $          1 .OR. ires_opt_hill .EQ. 1)) THEN
                   IF (conv_asr .EQ. 1) THEN
                      IF (ssm_env .EQ. 'TRUE ') THEN
                         dellec = dellec + wss351 * 1.9835
                      ELSE
                         IF (wcaenv(iwca3a) .EQ. 'FALSE' .OR. nnrctwca3a
     $                   .EQ. 'FALSE' .OR. ntarg .EQ. 0) THEN
                            dellec = dellec + amin1(wss351,dmnd_lec) *
     $                      1.9835
                         ENDIF
                      ENDIF
                   ENDIF
                ELSE
                   IF (ssm_env .EQ. 'TRUE ') THEN
                      dellec = dellec + wss351 * 1.9835
                   ELSE
                      IF (wcaenv(iwca3a) .EQ. 'FALSE' .OR. nnrctwca3a
     $                .EQ. 'FALSE' .OR. ntarg .EQ. 0) THEN
                         dellec = dellec + amin1(wss351,dmnd_lec) *
     $                   1.9835
                      ENDIF
                   ENDIF
                ENDIF
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

```
            ENDIF
            emerg_bypass_sta34 = emerg_bypass_sta34 + amin1
     $      (total_inflow - qinflw(indxsta(is7bsn-6),ncount(indxsta
     $      (is7bsn-6))), flow_lower_reach_max) - amax1
     $      (flow_above_capac -flow_through_s7,0.0)
            emerg_bypass_s7 = amin1(total_inflow - qinflw(indxsta
     $      (is7bsn-6),ncount(indxsta(is7bsn-6))), flow_lower_reach_max)
     $      - amax1(flow_above_capac -flow_through_s7,0.0)

            IF (ntarg_hill .GT.  0.AND. (iasr_opt_eaa(is7bsn-6) .EQ. 1
     $      .OR. ires_opt_hill .EQ. 1) .AND. conv_asr .EQ. 1 .OR.
     $      ((iasr_opt_eaa(is7bsn-6) .NE. 1 .AND. ires_opt_hill .NE. 1)
     $      .OR. ntarg_hill .EQ. 0)) THEN
               ws_to_lec_s351 = ws_to_lec_s351 + amin1(wss351,ws_to_lec)
               ws_to_lec_s351_nnrc = ws_to_lec_s351_nnrc + amin1(wss351,
     $         amax1(ws_to_lec-dmnd_lec_sa3_s150,0.0))
               if (dmnd_lec_sa3 .gt. 0.00001) then
                  rlok_flow_to_enp_s150 =amin1(ws_to_lec,
     $            dmnd_lec_sa3_s150) /dmnd_lec_sa3 * frf
               else
                  rlok_flow_to_enp_s150 = 0.0
               endif
               rlok_flow_to_enp = rlok_flow_to_enp_s150
     $         + rlok_flow_to_enp_s354
               ws_to_lec_s351_wca3a = ws_to_lec_s351_wca3a + amax1(amin1
     $         (ws_to_lec,dmnd_lec_sa3_s150)-rlok_flow_to_enp_s150,0.0)
               IF (flow_to_wca_prior .EQ. 'LEC') THEN
                  flow_to_import = flow_to_import + flow_thru_to_glades
                  flow_to_import_nnrc = flow_thru_to_glades
               ELSE IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $         dmnd_lec .AND. total_inflow_prescap_unadj.LE.dmnd_lec)
     $         THEN
                  flow_to_import = flow_to_import + rmax_to_glades
                  flow_to_import_nnrc = rmax_to_glades
               ELSE
                  flow_to_import = flow_to_import + amax1(flow_through -
     $            ws_to_lec , 0.0)
                  flow_to_import_nnrc = amax1(flow_through - ws_to_lec,
     $            0.0)
               ENDIF
            ENDIF
            flow_from_eaa_lok_s7 = amin1(excnnr,flow_lower_reach_max)
c       print *,ws_to_lec_s351_nnrc,ws_to_lec,rmax_to_glades
c    $     ,flow_thru_prescap,flow_to_import_nnrc,' CCCC  '
c    $     ,kflo(ifls7),kflo(ifls150),dmnd_lec_sa3_s150
c    $     ,excnnr,dmnd_lec_sa2_s7,runoff,flow_through,month,iday
c    $     ,flow_lower_reach_max,s7_new_hw,s7_tailw
         ELSE
c
c   proposed s7(s7new) is not pumping into STA: runoff from miami canal
c   basin not  backpumped into LOK is less than LEC demands,thus runoff
c   bypasses STA to meet the LEC demands
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (eaa_conv_opt_s7 .EQ. 'PUMP') THEN
               eaa_convey_max = eaa_convey(5,1)
            ELSE
```

```
            eaa_convey_max = eaa_convey(3,1)
         ENDIF
         flow_lower_reach_max = eaa_convey_max
         IF (igravity_s7 .EQ. 1) THEN
            s7capacity = s7_cap
         ELSE
            s7capacity = s7max
         ENDIF
         avail_cap_for_sta34_nnr = total_inflow
         IF(runoff .GT. eaa_convey_max) THEN
            IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
     $         (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ.
     $         idn_wca_indx(is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6)
     $         .NE. 'PIPEL  ') THEN

               KFLO(IFLS7) = eaa_convey_max + dischg_to_s7_frm_sta34

               IF (kflo(ifls7) .GT. s7capacity) THEN
                  flow_above_capac = kflo(ifls7) - s7capacity
                  kflo(ifls7) = s7capacity
               ELSE
                  flow_above_capac = 0.0
               ENDIF
               flow_from_eaa_lok_s7 = amax1(eaa_convey_max
     $         -flow_above_capac ,0.0)
            ELSE
               KFLO(IFLS7) = amin1(eaa_convey_max,dmnd_lec_sa2_s7) +
     $         dischg_to_s7_frm_sta34
               IF (opt_pipeline_ws(is7bsn-6) .EQ. 'PIPEL  ') THEN
                  ws_pipeline_2a = ws_pipeline_2a + amin1
     $            (eaa_convey_max, dmnd_lec_sa2_s7)
               ENDIF
               rem_runoff = eaa_convey_max - amin1(eaa_convey_max ,
     $         dmnd_lec_sa2_s7)
               KFLO(IFLS150) = amin1(rem_runoff,dmnd_lec_sa3_s150)
               IF (opt_pipeline_ws(is150bws-6) .EQ. 'PIPEL  ') THEN
                  ws_pipeline_3a_s150 = ws_pipeline_3a_s150 + amin1
     $            (rem_runoff,dmnd_lec_sa3_s150)
               ENDIF
               rem_runoff = rem_runoff - amin1(rem_runoff,
     $         dmnd_lec_sa3_s150)
               KFLO(IFLS7) = KFLO(IFLS7) + rem_runoff
               IF (opt_pipeline_ws(is7bsn-6) .EQ. 'PIPEL  ') THEN
                  ws_pipeline_2a = ws_pipeline_2a + rem_runoff
               ENDIF
               IF (kflo(ifls7) .GT. s7capacity) THEN
                  flow_above_capac = kflo(ifls7) - s7capacity
                  kflo(ifls7) = s7capacity
               ELSE
                  flow_above_capac = 0.0
               ENDIF
               flow_from_eaa_lok_s7 = amin1(eaa_convey_max,
     $         dmnd_lec_sa2_s7) + rem_runoff - flow_above_capac
               flow_from_eaa_lok_s7 = amax1(flow_from_eaa_lok_s7,0.0)
            ENDIF
            runoff_to_lec_s7 = flow_from_eaa_lok_s7
c        print *,runoff,rem_runoff,KFLO(IFLS150),KFLO(IFLS7)
```

```
c    $     ,dischg_to_s7_frm_sta34,dmnd_lec_sa2_s7,' XXX'
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               IF (ntarg_hill.GT. 0.AND. (iasr_opt_eaa(is7bsn-6) .EQ.
     $          1 .OR. ires_opt_hill .EQ. 1) .AND. conv_asr.EQ. 1.OR.
     $         ((iasr_opt_eaa(is7bsn-6) .NE.  1.AND. ires_opt_hill .NE.
     $         1) .OR. ntarg_hill .EQ. 0)) THEN
                  IAGFLO(IAG(IS7BSN),1) = NINT(runoff-eaa_convey_max)
     $            +NINT(flow_above_capac) + IAGFLO(IAG(IS7BSN),1)
               ENDIF
            ELSE
               flow_s7new_to_sta = 0.0
               IF ((iconvey_canal_no(is150bws-6) .EQ. iconvey_canal_no
     $         (is8bsn-6) .AND. idn_wca_indx (is150bws-6) .EQ.
     $         idn_wca_indx(is8bsn-6)) .AND. opt_pipeline_ws(is150bws-6)
     $         .NE. 'PIPEL  ') THEN

                  KFLO(IFLS7) = runoff + dischg_to_s7_frm_sta34
                  IF (kflo(ifls7) .GT. s7capacity) THEN
                     flow_above_capac = kflo(ifls7) - s7capacity
                     kflo(ifls7) = s7capacity
                  ELSE
                     flow_above_capac = 0.0
                  ENDIF
                  flow_from_eaa_lok_s7 = amax1(runoff-flow_above_capac,
     $            0.0)
               ELSE
                  KFLO(IFLS7) = amin1(runoff,dmnd_lec_sa2_s7) +
     $            dischg_to_s7_frm_sta34
                  IF (opt_pipeline_ws(is7bsn-6) .EQ. 'PIPEL  ') THEN
                     ws_pipeline_2a = ws_pipeline_2a + amin1(runoff ,
     $               dmnd_lec_sa2_s7)
                  ENDIF
                  rem_runoff = runoff - amin1(runoff ,dmnd_lec_sa2_s7)

                  KFLO(IFLS150) = amin1(rem_runoff,dmnd_lec_sa3_s150)
                  IF (opt_pipeline_ws(is150bws-6) .EQ. 'PIPEL  ') THEN
                     ws_pipeline_3a_s150 = ws_pipeline_3a_s150 + amin1
     $               (rem_runoff,dmnd_lec_sa3_s150)
                  ENDIF
                  rem_runoff = rem_runoff-amin1(rem_runoff,
     $            dmnd_lec_sa3_s150)
                  KFLO(IFLS7) = KFLO(IFLS7) + rem_runoff
                  IF (opt_pipeline_ws(is7bsn-6) .EQ. 'PIPEL  ') THEN
                     ws_pipeline_2a = ws_pipeline_2a + rem_runoff
                  ENDIF
                  IF (kflo(ifls7) .GT. s7capacity) THEN
                     flow_above_capac = kflo(ifls7) - s7capacity
                     kflo(ifls7) = s7capacity
                  ELSE
                     flow_above_capac = 0.0
                  ENDIF
               ENDIF
c        print *,runoff,rem_runoff,KFLO(IFLS150),KFLO(IFLS7)
c    $     ,dischg_to_s7_frm_sta34,dmnd_lec_sa2_s7,' XXX'
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
               flow_from_eaa_lok_s7 = amin1(runoff,dmnd_lec_sa2_s7) +
     $         rem_runoff - flow_above_capac
```

```
                flow_from_eaa_lok_s7 = amax1(flow_from_eaa_lok_s7,0.0)
                runoff_to_lec_s7 = flow_from_eaa_lok_s7
                IAGFLO(IAG(IS7BSN),1) = NINT(flow_above_capac) + IAGFLO
     $          (IAG(IS7BSN),1)
             ENDIF
          ENDIF
          excnnr = flow_from_eaa_lok_s7
c         write(*,246) total_inflow,runoff,flow_through,kflo(ifls2)
c    $,   kflo(ifls7),amax1(alocnnr,0.0)/86400.,s7spmx,dmnd_lec
c    $,   excess_flow,iagflo(iag(is7bsn),1),month,iday,ifyr+iyear-1
c    $,   qinflw(indxsta(is7bsn-6),ncount(indxsta(is7bsn-6)))
c
c    runoff from s-6 basin into sta-2
c
      ELSE IF (istopt .EQ. 3) THEN
          hillexc = 0.0
          ntarg = 0
          DO itarg = 1,no_targ_loc(is6bsn-6)
             IF (lwslo(itarg_area(itarg,is6bsn-6))) THEN
                ntarg = ntarg + 1
             ENDIF
          END DO
          ncount(indxsta(is6bsn-6)) = ncount(indxsta(is6bsn-6)) + 1
          IF (ifacteaa .EQ. 1) THEN
             flow_thru_prescap = amax1(fllks6_prescap - kfloref
     $       (is6bsn-6) ,0.0)
             flow_thru_prescap = amin1(flow_thru_prescap,flow_through)
             total_inflow_prescap = runoff + flow_thru_prescap
          ELSE
             total_inflow_prescap = total_inflow
          ENDIF
          IF ((total_inflow_prescap .GT. dmnd_lec .OR. (ifacteaa .EQ. 1
     $    .AND. total_inflow .GT. dmnd_lec)).OR.(((LWSLO_floor(iwca1)
     $    .AND. LWSLECSTA(is6bsn-6).AND.floor_grid_cell_opt.eq.'MARSH')
     $    .OR. rmin_stg_wca1 .gt.rmnstg(iwca1,iconvey_canal_no_wca
     $    (iwca1))+0.001)).AND. iopt_ws_lec_to_sta_wca_abv_flr .eq. 1)
     $    .OR. flow_to_wca_prior .EQ.'LEC') THEN
c         IF (total_inflow_prescap .GT. dmnd_lec .OR. (ifacteaa .EQ. 1
c    $    .AND. total_inflow .GT. dmnd_lec) .OR. flow_to_wca_prior .EQ.
c    $    'LEC') THEN
             oper_capac = cnlcap(ihill) - (1.0-pct_des(ihill,iseas))
     $       *cnlcap(ihill)/rcpfacteaa(is6bsn-6)
             IF (.NOT. DRYSEAS .AND. nint(runoff) .GT. 0) THEN
                rmax_to_glades = amin1(fllks6,oper_capac) - kfloref
     $          (is6bsn-6)
             ELSE
                rmax_to_glades = amin1(fllks6-kfloref(is6bsn-6),
     $          oper_capac)
             ENDIF
             rmax_to_glades = amax1(rmax_to_glades,0.0)
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             IF (flow_to_wca_prior .EQ. 'LEC') THEN
                IF (ntarg .GT. 0) THEN
                   avail_flow_to_lec = frac_lec(is6bsn-6)*total_inflow
                ELSE
                   avail_flow_to_lec = total_inflow
```

```
            ENDIF
            flow_to_lec = amin1(avail_flow_to_lec,dmnd_lec)
            rem_to_glades = total_inflow - flow_to_lec
            flow_thru_to_glades_bfop = amax1(rem_to_glades-runoff,
$           0.0)
            flow_thru_to_glades = amin1(flow_thru_to_glades_bfop ,
$           rmax_to_glades)
            total_inflow = amin1(runoff,rem_to_glades) +
$           flow_thru_to_glades
            flow_through = flow_through - (flow_thru_to_glades_bfop -
$           flow_thru_to_glades)
         ELSE
            IF (nasr_to_lok(1) .EQ. 0) THEN
               call bpts(iday,month,10.0,1,n_fc_zones+4,1,
$              stgref_asr_rec)
               IF (stagelo .LT. stgref_asr_rec .AND.(climate_index
$              (month) .LT. 1.5 .OR. ((itrib_hydro_cond_opt .EQ. 1
$              .AND. trib_rf_et .LT. -1.0 .AND.
$              s65e_runff_wkly .LT. 1500.) .OR.
$              itrib_hydro_cond_opt .EQ. 0)) .AND. DRYSEAS) THEN
                  frac_env_deliv = 0.00
               ELSE
                  frac_env_deliv = 1.00
               ENDIF
            ELSE
               frac_env_deliv = 1.00
            ENDIF
            frac_env_deliv = amin1(frac_env_deliv,frac_env_met)
            IF (ifacteaa .EQ. 1 .AND. total_inflow .GT. dmnd_lec
$           .AND. total_inflow_prescap.LE.dmnd_lec) THEN
               flow_thru_glades = amax1(flow_through
$              -flow_thru_prescap ,0.0)
               rmax_to_glades = amin1(rmax_to_glades,
$              flow_thru_glades) * frac_rem_capac(is6bsn-6) *
$              frac_env_deliv
               total_inflow = rmax_to_glades
            ELSE
               flow_through = amin1(flow_through,rmax_to_glades) *
$              frac_rem_capac(is6bsn-6) * frac_env_deliv
               total_inflow = runoff + flow_through
            ENDIF
         ENDIF
         call match (name_res_spec_code(22),1,6,resname,ntotres,6
$         ,imatch_sta2m)
         call match (name_res_spec_code(20),1,6,resname,ntotres,6
$         ,imatch_sta2w)
         total_volwtr_sta2 = volwtr (indxsta(is6bsn-6))
         if (imatch_sta2m .gt. 0) then
            total_volwtr_sta2 = total_volwtr_sta2
$           + volwtr(imatch_sta2m)
         endif
         if (imatch_sta2w .gt. 0) then
            total_volwtr_sta2 = total_volwtr_sta2
$           + volwtr(imatch_sta2w)
         endif
         qinflw(indxsta(is6bsn-6),ncount(indxsta(is6bsn-6)))= amin1
$        (total_volwtr_sta2,total_inflow,s6max*pctwma
```

```
     $      (indxsta(is6bsn -6)))
            flow_to_sta2_frm_eaa = amin1(qinflw(indxsta(is6bsn-6) ,
     $      ncount(indxsta(is6bsn-6))),runoff)
            IF (ifacteaa .EQ. 1 .AND. total_inflow .GT. dmnd_lec .AND.
     $      total_inflow_prescap.LE.dmnd_lec) flow_to_sta2_frm_eaa = 0.0
            kflo(ifls6) = qinflw(indxsta(is6bsn-6),ncount(indxsta
     $      (is6bsn-6)))
c
            res_area_sta2e = res_area(indxsta(is6bsn-6))
            if (imatch_sta2w .ne. 0) then
               res_area_sta2w = res_area(imatch_sta2w)
            else
               res_area_sta2w = 0.0
            endif
            if (imatch_sta2m .ne. 0) then
               res_area_sta2m = res_area(imatch_sta2m)
            else
               res_area_sta2m = 0.0
            endif
            sum_res_area_sta2 = res_area_sta2e+res_area_sta2w
     $      +res_area_sta2m
c
            ratio_flow_s6_sta2e = res_area_sta2e
     $      /sum_res_area_sta2
            if (imatch_sta2w .ne. 0) then
               ratio_flow_s6_sta2w = res_area(imatch_sta2w)
     $         /sum_res_area_sta2
            else
               ratio_flow_s6_sta2w = 0.0
            endif
            if (imatch_sta2m .ne. 0) then
               ratio_flow_s6_sta2m = res_area(imatch_sta2m)
     $         /sum_res_area_sta2
            else
               ratio_flow_s6_sta2m = 0.0
            endif
c
c    compute excess runoff from s-6 basin sta2 could not take
c
            rem_runoff_over_des_news6 = amax1(total_inflow - s6max *
     $      pctwma(indxsta(is6bsn-6)),0.0)
            rem_runoff_over_stor_sta2 = amax1(total_inflow -
     $      total_volwtr_sta2,0.0)
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            volwtr(indxsta(is6bsn-6))=volwtr(indxsta(is6bsn-6)) - qinflw
     $      (indxsta(is6bsn-6),ncount(indxsta(is6bsn-6)))
     $      * ratio_flow_s6_sta2e
            if (imatch_sta2w .ne. 0) then
              volwtr(imatch_sta2w) = volwtr(imatch_sta2w)
     $        - qinflw(indxsta(is6bsn-6),ncount(indxsta(is6bsn-6)))
     $        * ratio_flow_s6_sta2w
            endif
            if (imatch_sta2m .ne. 0) then
              volwtr(imatch_sta2m) = volwtr(imatch_sta2m)
     $        - qinflw(indxsta(is6bsn-6),ncount(indxsta(is6bsn-6)))
     $        * ratio_flow_s6_sta2m
```

```
            endif
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               hillexc = total_inflow - qinflw(indxsta(is6bsn-6) ,ncount
     $         (indxsta(is6bsn-6))) + flow_to_lec
     $         - amin1(flow_to_lec,flow_through) * (1.0
     $         - frac_del_frm_wca(iwca1))
               ws_to_lec = amin1(flow_to_lec,flow_through)
     $         * frac_del_frm_wca(iwca1)
            ELSE
               IF (ifacteaa.EQ. 1 .AND. total_inflow_unadj .GT. dmnd_lec
     $         .AND. total_inflow_prescap.LE.dmnd_lec) THEN
                  hillexc= total_inflow - qinflw(indxsta(is6bsn-6) ,
     $            ncount(indxsta(is6bsn-6))) + total_inflow_prescap
     $            - flow_thru_prescap*(1.0-frac_del_frm_wca(iwca1))
                  flow_through = total_inflow + flow_thru_prescap
     $            * frac_del_frm_wca(iwca1)
                  ws_to_lec = flow_thru_prescap*frac_del_frm_wca(iwca1)
               ELSE
                  hillexc= total_inflow - qinflw(indxsta(is6bsn-6) ,
     $            ncount(indxsta(is6bsn-6)))
                  ws_to_lec = 0.0
               ENDIF
            ENDIF
            capac_bypass = amin1(s6max - qinflw(indxsta(is6bsn-6) ,
     $      ncount(indxsta(is6bsn-6))),450.0)
            IF (hillexc .GT. capac_bypass) THEN
               iagflo(iag(is7bsn),1) = iagflo(iag(is7bsn),1) + nint
     $         (amax1(hillexc-flow_through-capac_bypass,0.0))
               flow_through = flow_through - amin1(hillexc
     $         -capac_bypass ,flow_through)
               ws_to_lec = ws_to_lec - amin1(hillexc-capac_bypass,
     $         ws_to_lec)
            ENDIF
            IF (opt_pipeline_ws(is6bsn-6) .EQ. 'PIPEL  ') THEN
               ws_pipeline_ca1 = ws_pipeline_ca1 + amin1(hillexc ,
     $         capac_bypass,dmnd_lec)
            ENDIF
            kflo(ifls2) = kflo(ifls2) + flow_through
            kfloref(is6bsn-6) = kfloref(is6bsn-6) + flow_through
            wss351 = wss351 + flow_through
            IF (ssm_env .EQ. 'TRUE ') THEN
               dellec = dellec + flow_through * 1.9835
            ELSE
               IF (wcaenv(iwca2a) .EQ. 'FALSE' .OR. ntarg .EQ. 0) THEN
                  dellec = dellec + amin1(flow_through,dmnd_lec) *
     $            1.9835
               ENDIF
            ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            ws_to_lec_s351 = ws_to_lec_s351 + amin1(flow_through,
     $      ws_to_lec)
            ws_to_lec_s351_hill = ws_to_lec_s351_hill + amin1
     $      (flow_through,ws_to_lec)
            IF (opt_for_hill_bypass_runoff .EQ. 'TRUE ') THEN
               ws_from_lo_s6 = amin1(s6max,hillexc)
               IF (opt_pipeline_ws(is6bsn-6) .NE. 'PIPEL  ') THEN
                  qu(ica1) = qu(ica1) + ws_from_lo_s6 * 86400.
```

```
               ENDIF
               runoff_to_lec_s6 = amin1(hillexc,s6max,dmnd_lec) -
     $         flow_through
               runoff_to_lec_s6 = amax1(runoff_to_lec_s6,0.0)
               flow_to_s7_sta2byp = 0.0
            ELSE
               flow_to_s7_sta2byp = amin1(total_inflow - qinflw(indxsta
     $         (is6bsn-6),ncount(indxsta(is6bsn-6))) ,capac_bypass) -
     $         dmnd_lec
               flow_to_s7_sta2byp = amax1(flow_to_s7_sta2byp,0.0)
               ws_from_lo_s6 = amin1(flow_to_s7_sta2byp,dmnd_lec)
               runoff_to_lec_s6 = amax1(ws_from_lo_s6-flow_through,0.0)
               hillexc = ws_from_lo_s6
               call match(name_res_spec_code(20),1,6,resname,ntotres+ 1,
     $         6,ires_no_sta2)
               DO idns = 1,ndspts(ires_no_sta2,1)
                  pond(nodepw(ires_no_sta2,1,idns)) = pond(nodepw
     $            (ires_no_sta2,1,idns))+flow_to_s7_sta2byp*86400. /
     $            (ndspts(ires_no_sta2,1)*gdar)
               END DO
c        print *,flow_to_s7_sta2byp,kflo(ifls7),'  AFT FLOW FRM STA2'
c    $      ,dischg_to_s7_frm_sta34
            ENDIF
c      print *,ws_from_lo_s6,flow_through,kflo(ifls2),'  in flowtostas'
            IF (flow_to_wca_prior .EQ. 'LEC') THEN
               flow_to_import = flow_to_import + flow_thru_to_glades
               flow_to_import_hill = flow_thru_to_glades
            ELSE IF (ifacteaa .EQ. 1 .AND. total_inflow_unadj .GT.
     $      dmnd_lec .AND. total_inflow_prescap.LE.dmnd_lec) THEN
               flow_to_import = flow_to_import + rmax_to_glades
               flow_to_import_hill = rmax_to_glades
            ELSE
               flow_to_import = flow_to_import + flow_through
               flow_to_import_hill = flow_through
            ENDIF
         ELSE
            hillexc = runoff
            flow_to_sta2_frm_eaa = 0.0
            runoff_to_lec_s6 = runoff
         ENDIF
c        write(*,246) total_inflow,runoff,flow_through,kflo(ifls2)
c    $,   kflo(ifls6),dmnd_hill,s6max,dmnd_lec
c    $,   hillexc,iagflo(iag(is7bsn),1),month,iday,ifyr+iyear-1
c    $,   qinflw(indxsta(is6bsn-6),ncount(indxsta(is6bsn-6)))
c
c     runoff from 298 districts into sta34,et al
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      ELSE IF (istopt .EQ. 4) THEN
        Q1 = frac_298r_s(1)*total_inflow
        Q2 = frac_298r_s(2)*total_inflow
        Q3 = frac_298r_s(3)*total_inflow
        runoff_715frms = frac_298r_s(4)*total_inflow
        CAPAC = 1160.0
        do ieaab = 1,neaacnl
           add_conv_red(ieaab) = 0.0
        enddo
```

```
c
c  take into account water supply from EAA reservoir(s)
c  to meet demands within EAA basin
c
         do irs = 1,nres_ws
          if (iconv_use_opt_ws(irs) .eq. 1) then
            do js = 1,noutlets(irs)
              if (ieaa_bsn(irs,js) .gt. 0) then
                add_conv_red(ieaa_bsn(irs,js)) =
     $          add_conv_red(ieaa_bsn(irs,js)) +
     $          res_outflow_to_basin(irs,js)
              endif
            enddo
          endif
         enddo
c
        flo_eaa_cnl(is8bsn-6) = runoff_south(is8bsn-6)+kflo(ifls3)
     $   + add_conv_red(is8bsn-6)
        flo_eaa_cnl(is6bsn-6) = runoff_south(is6bsn-6)+divers
     $   + dmnd_hill + max(add_conv_red(is7bsn-6) - dmnd_nnr_orig
     $   + dmnd_nnr_aft_resasr,0.0)
        flo_eaa_cnl(is5absn-6)= runoff_south(is5absn-6) + kflo(iflhg5)
        rinf_struc_cap(is8bsn-6) = g372cap
        rinf_struc_cap(is6bsn-6) = s6max
        rinf_struc_cap(is7bsn-6) = g370cap
        do ires0 = 1,ntotres
           temp_avail_stor(ires0) = VOLWTR(ires0)
        enddo

c
c  To MIAMI and/or NNRC Canals
c
        if (istaopt(8) .eq.1) then
c
c  South Shore District
c
         ADDQ = AMIN1(Q1,VOLWTR(INDXSTA(8)) ,AMAX1(CNLCAP
     $   (idest_eaacnl_298d(1))-flo_eaa_cnl(idest_eaacnl_298d(1)),0.0)
     $   ,amax1(rinf_struc_cap(idest_eaacnl_298d(1))-flo_eaa_cnl
     $   (idest_eaacnl_298d(1)),0.0),struc_cap_298d_s(1))
         ADDQ = AMAX1(ADDQ,0.0)
c        print *,flo_eaa_cnl,Q1,CNLCAP(1),g372cap,struc_cap_298d_s(1)
c    $    ,' 298DMIAM  ',total_inflow,ADDQ
         runoff_south(idest_eaacnl_298d(1))=runoff_south
     $   (idest_eaacnl_298d(1))+addq
         flo_eaa_cnl(idest_eaacnl_298d(1))=flo_eaa_cnl
     $    (idest_eaacnl_298d(1)) + ADDQ
         temp_avail_stor(INDXSTA(8)) = temp_avail_stor(INDXSTA(8))
     $    - ADDQ
         sspsfcd_to_lok = min(Q1 - ADDQ,rmax_cap_298d_to_lok(1))
         if (Q1 - ADDQ .gt. rmax_cap_298d_to_lok(1)) then
            IAGFLO(IAG(ILAGBSN),1) = NINT(Q1 - ADDQ
     $       - rmax_cap_298d_to_lok(1)) + IAGFLO(IAG(ILAGBSN),1)
         endif
         sspsfcd_to_sta = ADDQ
       else
         sspsfcd_to_lok = min(Q1,rmax_cap_298d_to_lok(1))
```

```
          if (Q1 .gt. rmax_cap_298d_to_lok(1)) then
            IAGFLO(IAG(ILAGBSN),1) = NINT(Q1 -
     $      rmax_cap_298d_to_lok(1)) + IAGFLO(IAG(ILAGBSN),1)
          endif
        endif
C
C  S-236 Basin
c
        if (istaopt(8) .eq. 1) then
          flo_eaa_cnl_0 = flo_eaa_cnl(is8bsn-6)
          IF (DMDS236/1. 9835.LT. 0.0) THEN
             ADDQ = AMIN1(ABS(DMDS236)/1.9835,temp_avail_stor(INDXSTA(8))
     $       ,AMAX1(CNLCAP(1)-flo_eaa_cnl_0,0.0) ,amax1(g372cap-
     $       flo_eaa_cnl_0,0.0),504.0)
c            QINFLW(INDXSTA(8),1) = QINFLW(INDXSTA(8),1) + ADDQ
             DMDS236 = DMDS236 + ADDQ*1.9835
             S236SO = ADDQ
             s236lok = min(abs(DMDS236),255*1.9835)
             DMDS236 = -s236lok
          ELSE
             ADDQ = 0.0
             S236SO = 0.0
          ENDIF
c         print *,DMDS236/1. 9835,g372cap,CNLCAP(1),flo_eaa_cnl_0
c    $      ,ADDQ,temp_avail_stor(INDXSTA(8)),'  S236SO'
          runoff_south(is8bsn-6)=runoff_south(is8bsn-6)+ADDQ
          temp_avail_stor(INDXSTA(8))=temp_avail_stor(INDXSTA(8))-ADDQ
          flo_eaa_cnl(is8bsn-6)=flo_eaa_cnl(is8bsn-6)+ADDQ
C
C  G-136 Flow from Western Basin
c
          flo_eaa_cnl_1 = flo_eaa_cnl(is8bsn-6)
          ADDQ = AMIN1(KFLO(IG136),temp_avail_stor(INDXSTA(8))
     $   ,AMAX1(CNLCAP(1)-flo_eaa_cnl_1,0.0))
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
          KFLO(IG136) = KFLO(IG136) - ADDQ
          G136EAA = KFLO(IG136)
          g136_to_sta34 = ADDQ
          runoff_south(is8bsn-6) = runoff_south(is8bsn-6) + ADDQ
c
c         QINFLW(INDXSTA(8),1) = QINFLW(INDXSTA(8),1) + ADDQ
          temp_avail_stor(INDXSTA(8))=temp_avail_stor(INDXSTA(8))-ADDQ

c         VOLWTR(INDXSTA(8)) = VOLWTR(INDXSTA(8)) - ADDQ
        endif
C
C  ESWCD Runoff to HILL Canal
c
        if (istaopt(7) .eq.1) then
          ADDQ = AMIN1(Q2,temp_avail_stor(INDXSTA(7)),AMAX1(CNLCAP
     $   (idest_eaacnl_298d(2))-flo_eaa_cnl(idest_eaacnl_298d(2))
     $   ,0.0),amax1(rinf_struc_cap(idest_eaacnl_298d(2))-flo_eaa_cnl
     $   (idest_eaacnl_298d(2)),0.0),struc_cap_298d_s(2))
          ADDQ = AMAX1(ADDQ,0.0)
          runoff_south(idest_eaacnl_298d(2))=runoff_south
     $   (idest_eaacnl_298d(2)) + ADDQ
          eswcd_to_lok = min(Q2 - ADDQ,rmax_cap_298d_to_lok(2))
```

```
          if (Q2 - ADDQ .gt. rmax_cap_298d_to_lok(2)) then
             IAGFLO(IAG(ILAGBSN),1) = NINT(Q2 - ADDQ
     $        - rmax_cap_298d_to_lok(2)) + IAGFLO(IAG(ILAGBSN),1)
          endif
          eswcd_to_sta = ADDQ
          flo_eaa_cnl(idest_eaacnl_298d(2))=flo_eaa_cnl
     $    (idest_eaacnl_298d(2)) + ADDQ
          temp_avail_stor(INDXSTA(7))=temp_avail_stor(INDXSTA(7))
     $    -ADDQ
c         VOLWTR(INDXSTA(7)) = VOLWTR(INDXSTA(7)) - ADDQ
        else
          eswcd_to_lok = min(Q2,rmax_cap_298d_to_lok(2))
          if (Q2 .GT. rmax_cap_298d_to_lok(2)) then
            IAGFLO(IAG(ILAGBSN),1) = NINT(Q2 -
     $      rmax_cap_298d_to_lok(2)) + IAGFLO(IAG(ILAGBSN),1)
          endif
        endif
c
c   715 (Cloister Farms) to HILL
c
         if (istaopt(7) .eq.1) then
           QCLOISTER = min(runoff_715frms,struc_cap_298d_s(4))
           ADDQ = AMIN1(QCLOISTER,temp_avail_stor(INDXSTA(7)),
     $     AMAX1(CNLCAP(idest_eaacnl_298d(4))-flo_eaa_cnl
     $     (idest_eaacnl_298d(4)),0.0),amax1(rinf_struc_cap
     $     (idest_eaacnl_298d(4))-flo_eaa_cnl(idest_eaacnl_298d(4)),0.0))
           runoff_south(idest_eaacnl_298d(4))
     $     =runoff_south(idest_eaacnl_298d(4))
     $     +ADDQ
           q715_farms_to_lok = min(runoff_715frms-addq,
     $     rmax_cap_298d_to_lok(4))
           if (runoff_715frms-addq .gt. rmax_cap_298d_to_lok(4)) then
              IAGFLO(IAG(ILAGBSN),1) = NINT(runoff_715frms-addq
     $         - rmax_cap_298d_to_lok(4)) + IAGFLO(IAG(ILAGBSN),1)
           endif
           q715_farms_to_sta = ADDQ
           flo_eaa_cnl(idest_eaacnl_298d(4))=flo_eaa_cnl
     $     (idest_eaacnl_298d(4))+ADDQ
           temp_avail_stor(INDXSTA(7))=temp_avail_stor(INDXSTA(7))
     $     -ADDQ
         else
           q715_farms_to_lok=min(runoff_715frms,rmax_cap_298d_to_lok(4))
           if (runoff_715frms .gt. rmax_cap_298d_to_lok(4)) then
               IAGFLO(IAG(ILAGBSN),1) = NINT(runoff_715frms-
     $         rmax_cap_298d_to_lok(4)) + IAGFLO(IAG(ILAGBSN),1)
           endif
         endif
c
c  EBWCD Runoff to WPB Canal
c
        if (istaopt(9) .eq. 1) THEN
           ADDQ = AMIN1(Q3,AMAX1(CNLCAP(idest_eaacnl_298d(3))
     $     -flo_eaa_cnl(idest_eaacnl_298d(3)),0.0),struc_cap_298d_s(3))
           kflo(ifls5a) = kflo(ifls5a) + ADDQ
           kflo(ifls5a5) = kflo(ifls5a5) + ADDQ
           eaa_runff_s5a_to_sta = eaa_runff_s5a_to_sta + ADDQ
           s5a2_so=s5a2_so + ADDQ
```

```
          ebwcd_to_lok = min(Q3 - ADDQ,rmax_cap_298d_to_lok(3))
          if (Q3 - ADDQ .gt. rmax_cap_298d_to_lok(3)) then
             IAGFLO(IAG(ILAGBSN),1) = NINT(Q3 - ADDQ
     $        - rmax_cap_298d_to_lok(3)) + IAGFLO(IAG(ILAGBSN),1)
          endif
          ebwcd_to_sta = ADDQ
       else
          ebwcd_to_lok = min(Q3,rmax_cap_298d_to_lok(3))
          IF (Q3 .GT. rmax_cap_298d_to_lok(3)) THEN
            IAGFLO(IAG(ILAGBSN),1) = NINT(Q3-rmax_cap_298d_to_lok(3))
     $      + IAGFLO(IAG(ILAGBSN),1)
          ENDIF
       endif
c
       IF (nint(total_inflow) .GT. 0) THEN
          KFLO(IFLAGQ) = - (ebwcd_to_lok+q715_farms_to_lok
     $    +eswcd_to_lok+sspsfcd_to_lok)
       endif
c
      ELSE IF (istopt .EQ. 5) THEN
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------
C     ROUTE DESIRED PERCENTAGE of G155,G-89,AND G-88 FLOWS INTO STA-5
C-----------------------------------------
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (ISTAOPT(6) .EQ. 1) THEN
            rem_runoff_over_stor_sta5 = 0.0
            IF (ic139_rnff_fix_split_opt .eq. 0) then
              TOTFLOW = amax1(KFLO(IG155),0.0) + KFLO(IG89) + KFLO(IG88)

              DO J = 1,NINSTR(indxsta(6))
                 QINFLW(indxsta(6),J) = AMIN1(TOTFLOW/NINSTR(indxsta(6))
     $           ,VOLWTR(indxsta(6))) + QINFLW(indxsta(6),J)
                 rem_runoff_over_stor_sta5 = amax1(totflow/NINSTR
     $           (indxsta(6))-VOLWTR(indxsta(6)) ,0.0) +
     $           rem_runoff_over_stor_sta5
                 VOLWTR(indxsta(6)) = VOLWTR(indxsta(6)) -
     $           amin1(TOTFLOW/NINSTR(indxsta(6)),VOLWTR(indxsta(6)))
              END DO
c          print *,TOTFLOW,VOLWTR(indxsta(6)),rem_runoff_over_stor_sta5
            ELSE

              TOTFLOW = (amax1(KFLO(IG155),0.0)+KFLO(IG89)+KFLO(IG88))
     $        * frac_c139basin_sta5
              DO J = 1,NINSTR(indxsta(6))
                QINFLW(indxsta(6),J) = AMIN1(TOTFLOW/NINSTR(indxsta(6))
     $          ,VOLWTR(indxsta(6))) + QINFLW(indxsta(6),J)
                 rem_runoff_over_stor_sta5 = amax1(totflow/NINSTR
     $           (indxsta(6))-VOLWTR(indxsta(6)) ,0.0) +
     $           rem_runoff_over_stor_sta5
                 VOLWTR(indxsta(6)) = VOLWTR(indxsta(6)) -
     $           amin1(TOTFLOW/NINSTR(indxsta(6)),VOLWTR(indxsta(6)))
              END DO
            ENDIF
         ENDIF
C-----------------------------------------
C     ROUTE RUNOFF FROM SUGAR CORP LAND TO STA-6
```

```
C-----------------------------------------
         IF (ISTAOPT(10) .EQ. 1) THEN
            sugh_runoff_over_des_strsta6 = 0.0
            rem_runoff_over_stor_sta6 = 0.0
            SUGRNFF = AMIN1(DMDSUGH,0.0) / 1.9835
            SUGRNFF = ABS(SUGRNFF)
            IF (NINSTR(indxsta(10)) .LT. 3) THEN
               IF (DRYSEAS) THEN
                  frac_runoff_c139 = 0.05
               ELSE
                  frac_runoff_c139 = 0.10
               ENDIF
               STR_INFLOW = frac_runoff_c139 * (amax1(KFLO(IG155),0.0) +
     $         KFLO(IG89) + KFLO(IG88))
               qu(is140up) = qu(is140up) + STR_INFLOW * 86400.
               inflow_to_sta6_frm_unit1 = str_inflow
            ENDIF
c
            DO J = 1,NINSTR(indxsta(10))
c
c       US SUGAR RANCH RUNOFF
c
               IF (j .EQ. 1) THEN
                  sugh_runoff_over_des_strsta6=amax1(sugrnff - weircff
     $            (indxsta(10),j,1),0.0)
                  STR_INFLOW = AMIN1(SUGRNFF,WEIRCFF(indxsta(10),J,1))
                  rem_runoff_over_stor_sta6 = rem_runoff_over_stor_sta6
     $            + amax1(STR_INFLOW-VOLWTR(INDXSTA(10)),0.0)
                  STR_INFLOW = AMIN1(STR_INFLOW,VOLWTR(INDXSTA(10)))
                  VOLWTR(INDXSTA(10)) = VOLWTR(INDXSTA(10)) - STR_INFLOW
                  QINFLW(indxsta(10),J) = STR_INFLOW+QINFLW(indxsta
     $            (10),J)
c
c  calculate excess flow sta-6 would not take
c
                  IAGFLO(IAG(ISUGBSN),1) = NINT(SUGRNFF-STR_INFLOW) +
     $            IAGFLO(IAG(ISUGBSN),1)
               ELSE IF (j .EQ. 2) THEN
c
c      fraction of C139 Basin runoff
c
                  IF (ic139_rnff_fix_split_opt .eq. 0) THEN
                    STR_INFLOW = rem_runoff_over_stor_sta5
                    rem_runoff_over_stor_sta6 = amax1(str_inflow -volwtr
     $              (indxsta(10)),0.0) + rem_runoff_over_stor_sta6
                    STR_INFLOW = AMIN1(STR_INFLOW,VOLWTR(INDXSTA(10)))
                    QINFLW(indxsta(10),J) = STR_INFLOW + QINFLW(indxsta
     $              (10),J)
                    VOLWTR(INDXSTA(10)) = VOLWTR(INDXSTA(10))-STR_INFLOW
                  ELSE
c
c      fraction of C139 Basin runoff
c
                    STR_INFLOW = frac_c139basin_sta6 * (KFLO(IG155)
     $              + KFLO(IG89) + KFLO(IG88))
                    rem_runoff_over_stor_sta6 = amax1(str_inflow -volwtr
     $              (indxsta(10)),0.0) + rem_runoff_over_stor_sta6
```

```
                    STR_INFLOW = AMIN1(STR_INFLOW,VOLWTR(INDXSTA(10)))
                    QINFLW(indxsta(10),J) = STR_INFLOW + QINFLW(indxsta
     $              (10),J)
                    VOLWTR(INDXSTA(10)) = VOLWTR(INDXSTA(10))-STR_INFLOW
                  ENDIF
               ELSE IF (j .EQ. 3) THEN
c
c    Runoff from C139 ANNEX
c
                  IF (DRYSEAS) THEN
                     frac_runoff_c139 = 0.05
                  ELSE
                     frac_runoff_c139 = 0.10
                  ENDIF
                  STR_INFLOW = frac_runoff_c139 * (amax1(KFLO(IG155),
     $            0.0) + KFLO(IG89) + KFLO(IG88))
                  rem_runoff_over_stor_sta6 = amax1(str_inflow - weircff
     $            (indxsta(10),j,1),0.0) + rem_runoff_over_stor_sta6
                  inflow_to_sta6_frm_unit1 = str_inflow
c
                  QINFLW(indxsta(10),J) = STR_INFLOW + QINFLW(indxsta
     $            (10),J)
                  VOLWTR(INDXSTA(10)) = VOLWTR(INDXSTA(10)) - STR_INFLOW
               ENDIF
            END DO
c
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
      ELSE IF (istopt .EQ. 6) THEN
CBW CBW
C-----------------------------------------------------------------
C    MAKEUP WATER FOR STA OPTION AS RESULT OF BMPS
C----------------------------------------------------------------
C
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (make_up_water_opt .EQ. 'MAKEUP ') THEN
C
c  calculate conveyance for miami canal for case of pumping
c  into STA3&4
c
            IF (istaopt(is8bsn-6) .EQ. 1) THEN
               ctw = pmp_int_new(is8bsn-6)
               ireach = 5
            ELSE
               ireach = 1
               ctw = pmp_int(is8bsn-6)
            ENDIF
            IF (.not. LFORPUMP(is8bsn-6)) THEN
              istype = 1
              istr_eaa = 5
              chw = stagelo
              rmax_cuhw = rmaxstg(is8bsn-6)
            ELSE
              istype = 2
              istr_eaa = 9
              chw = amin1(stagelo+rmax_lift(is8bsn-6),
```

```
     $        rmaxstg(is8bsn-6))
              rmax_cuhw = chw
           ENDIF
           delta0 = (chw - ctw) / 25.0
           call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo ,
     $     delta0,1.0,100,miamic_cap_upper_pump,s354cap,s354_tw ,
     $     ifacteaa)
           IF (s354_tw .GT. rmax_cuhw) THEN
              s354_tw = rmax_cuhw
              s354cap = -1.0
              call eaaspill(s354cap,stagelo,s354_tw,istr_eaa,
     $        ifacteaa)
              qcanal = -1.0
              IF (s354_tw.GT.ctw) THEN
                 call eaa_prof_lu_table(qcanal,s354_tw,ctw, ireach,
     $           ifacteaa)
              ELSE
                 qcanal = 0.0
              ENDIF
              miamic_cap_upper_pump = qcanal
           ENDIF
        ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
        grf3amx = miamic_cap_upper_pump
        s354mx = s354cap
c
c conveyance capacity for NNRC
c
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
        IF (istaopt(is7bsn-6) .EQ. 1) THEN
           ctw = pmp_int_new(is7bsn-6)
           ireach = 6
        ELSE
           ireach = 2
           ctw = pmp_int(is7bsn-6)
        ENDIF
        IF (.not. LFORPUMP(is7bsn-6)) THEN
           istype = 1
           istr_eaa = 6
           chw = stagelo
           rmax_cuhw = rmaxstg(is7bsn-6)
        ELSE
           istype = 2
           istr_eaa = 10
           chw = amin1(stagelo+rmax_lift(is8bsn-6),
     $     rmaxstg(is7bsn-6))
           rmax_cuhw = chw
        ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
        delta0 = (chw - ctw) / 25.0
        call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo ,
     $  delta0,1.0,100,nnrc_cap_upper_pump,s351cap,s351_tw ,
     $  ifacteaa)
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
        IF (s351_tw .GT. rmax_cuhw) THEN
           s351_tw = rmax_cuhw
           s351cap = -1.0
```

```
            call eaaspill(s351cap,stagelo,rmaxstg(is7bsn-6),istr_eaa,
     $      ifacteaa)
                qcanal = -1.0
                IF (s351_tw .GT. ctw) THEN
               call eaa_prof_lu_table(qcanal,s351_tw,ctw, ireach
     $             ,ifacteaa)
            ELSE
               qcanal = 0.0
            ENDIF
            nnrc_cap_upper_pump = qcanal
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         fllks6 = -1.0
         call eaa_prof_lu_table(fllks6,s351_tw,pmp_int(is6bsn-6) ,3,
     $   ifacteaa)
         grf2amx = nnrc_cap_upper_pump
         s351mx = s351cap
c
c     compute targets to be met on daily basis
c
         IF (IDAY .EQ. 1 .OR. makeup_water_restr .EQ. 'TRUE ')
     $      TARGNOTMT = 0.0
c
         IF (iday_lok_in_ssm .eq. 3 .and. bmp_ssm_opt .eq. 'YES') THEN
            TARGET = 0.0
         ELSE
            TARGET = BMPVOL/1. 9835* frac_env_met * frac_area_eaa_bmp
     $      + TARGNOTMT
         ENDIF

         TOTALCP = CNLCAP(imiami) + CNLCAP(innr) + CNLCAP(ihill)
     $   + CNLCAP(iwpb)
             TARGSTA3 = TARGET * CNLCAP(imiami) / TOTALCP
             TARGSTA4 = TARGET * CNLCAP(innr) / TOTALCP
             TARGSTA2 = TARGET * CNLCAP(ihill) / TOTALCP
             TARGSTA1 = TARGET * CNLCAP(iwpb) / TOTALCP
C
C      DETERMINE AMOUNT OF MAKEUP WATER TO BE ROUTED INTO STAS
C
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
     $       (idn_wca_indx(is8bsn-6)) .LE. regstg_wca(iwca3a,2) .AND.
     $   trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca
     $       (iwca2a,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6))
     $   .LE. regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ.
     $   'FALSE' .AND. trig_wca_stg(idn_wca_indx(is8bsn-6)) .LE.
     $       regstg_wca(iwca3a,2))) THEN
                IF (KFLO(IFLS3) .GE. 0.) THEN
               IF (istaopt(is8bsn-6) .EQ. 1) THEN
                  RMKVOLSTA3 = AMIN1(TARGSTA3,VOLWTR(INDXSTA
     $                (IS8BSN-6)),AMAX1(GRF3AMX-(runoff_south(is8bsn-6)
     $            +KFLO(IFLS3)),0.0),amax1(s354mx
     $            -kflo(ifls3),0.0),g372cap-qinflw(indxsta(is8bsn-6)
     $            ,1))
                    RMKVOLSTA3 = AMAX1(RMKVOLSTA3,0.0)
                  QINFLW(INDXSTA(IS8BSN-6),1) = QINFLW(INDXSTA
     $            (IS8BSN-6),1) +RMKVOLSTA3
```

```
c
                  KFLO(IFLS3) = KFLO(IFLS3) + RMKVOLSTA3
                  kfloref(is8bsn-6) = kfloref(is8bsn-6) + RMKVOLSTA3
                  flow_to_import = flow_to_import + RMKVOLSTA3
                  flow_to_import_miami = flow_to_import_miami
     $              + RMKVOLSTA3
c
                  VOLWTR(INDXSTA(IS8BSN-6)) = VOLWTR(INDXSTA
     $            (IS8BSN-6)) - RMKVOLSTA3
               ELSE
cbw check runoff_south split on lines
                  RMKVOLSTA3 = AMIN1(TARGSTA3 ,AMAX1(GRF3AMX -
     $              (runoff_south(is8bsn-6)+kflo(ifls3)),0.0) ,
     $            amax1(s354mx - kflo(ifls3),0.0),s8max-kflo(ifls8))
                  RMKVOLSTA3 = AMAX1(RMKVOLSTA3,0.0)
                  KFLO(IFLS3) = KFLO(IFLS3) + RMKVOLSTA3
                  KFLO(IFLS8) = KFLO(IFLS8) + RMKVOLSTA3
                  kfloref(is8bsn-6) = kfloref(is8bsn-6) + RMKVOLSTA3
                  flow_to_import = flow_to_import + RMKVOLSTA3
                  flow_to_import_miami = flow_to_import_miami +
     $              RMKVOLSTA3
               ENDIF
            ENDIF
         ELSE
            RMKVOLSTA3 = 0.0
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (istaopt(is7bsn-6) .EQ. 1) THEN
            IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
     $      (idn_wca_indx(is8bsn-6)) .LE.regstg_wca(iwca3a,2) .AND.
     $      trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca
     $      (iwca2a,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6))
     $      .LE. regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ.
     $      'FALSE' .AND. trig_wca_stg(idn_wca_indx(is8bsn-6)) .LE.
     $      regstg_wca(iwca3a,2))) THEN
                   IF (KFLO(IFLS2) .GE. 0.) THEN
                  RMKVOLSTA4 = AMIN1(TARGSTA4,VOLWTR(INDXSTA(IS7BSN
     $            -6)) ,AMAX1(GRF2AMX-QINFLW(INDXSTA(IS7BSN-6),2),
     $            0.0) - amax1(alocnnr,0.0)*pctmet/86400. , amax1
     $            (s351mx-kflo(ifls2),0.0), g370cap-qinflw(indxsta
     $            (is7bsn-6),2))
                  RMKVOLSTA4 = AMAX1(RMKVOLSTA4,0.0)
                  QINFLW(INDXSTA(IS7BSN-6),2) = QINFLW(INDXSTA
     $            (IS7BSN-6),2) +RMKVOLSTA4
                  KFLO(IFLS2) = KFLO(IFLS2) + RMKVOLSTA4
c
                  kfloref(is7bsn-6) = kfloref(is7bsn-6) + RMKVOLSTA4
                  flow_to_import = flow_to_import + RMKVOLSTA4
                  flow_to_import_nnrc = flow_to_import_nnrc +
     $            RMKVOLSTA4
c
                  VOLWTR(INDXSTA(IS7BSN-6)) = VOLWTR(INDXSTA
     $            (IS7BSN-6)) -RMKVOLSTA4
               ENDIF
            ELSE
               RMKVOLSTA4 = 0.0
            ENDIF
```

```
         ELSE
            IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
     $      (idn_wca_indx(is8bsn-6)) .LE.regstg_wca(iwca3a,2) .AND.
     $      trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca
     $      (iwca2a,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6))
     $      .LE. regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ.
     $      'FALSE' .AND. trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE.
     $      regstg_wca(iwca2a,1))) THEN
               IF (KFLO(IFLS2) .GE. 0.) THEN
cbw check runoff_south split on lines
                  RMKVOLSTA4 = AMIN1(TARGSTA4 ,AMAX1(GRF2AMX -
     $            runoff_south(is7bsn-6) - amax1(alocnnr,0.0)*pctmet/
     $            86400.,0.0) ,amax1(s351mx-kflo(ifls2),0.0) ,
     $            s7max-kflo(ifls7))
              RMKVOLSTA4 = AMAX1(RMKVOLSTA4,0.0)
              KFLO(IFLS2) = KFLO(IFLS2) + RMKVOLSTA4
              KFLO(IFLS7) = KFLO(IFLS7) + RMKVOLSTA4
c
              kfloref(is7bsn-6) = kfloref(is7bsn-6) + RMKVOLSTA4
              flow_to_import = flow_to_import + RMKVOLSTA4
              flow_to_import_nnrc = flow_to_import_nnrc +
     $            RMKVOLSTA4
               ENDIF
            ELSE
               RMKVOLSTA4 = 0.0
            ENDIF
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (istaopt(is6bsn-6) .EQ. 1) THEN
            IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
     $      (idn_wca_indx(is8bsn-6)) .LE.regstg_wca(iwca3a,2) .AND.
     $      trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca
     $      (iwca2a,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6))
     $      .LE. regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ.
     $      'FALSE' .AND. trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE.
     $      regstg_wca(iwca2a,1))) THEN
               IF (KFLO(IFLS2) .GE. 0.) THEN
                  floref = ws_from_lo_s6 + dmnd_hill + wsdel_to_sta
     $            (indxsta(is6bsn-6))
                  RMKVOLSTA2 = AMIN1(TARGSTA2,VOLWTR(INDXSTA(IS6BSN
     $            -6)) ,AMAX1(FLLKS6-KFLO(IFLS6),0.0) - floref, amax1
     $            (s351mx-kflo(ifls2),0.0),s6max-kflo(ifls6))
                  RMKVOLSTA2 = AMAX1(RMKVOLSTA2,0.0)
                  KFLO(IFLS2) = KFLO(IFLS2) + RMKVOLSTA2
                  KFLO(IFLS6) = KFLO(IFLS6) + RMKVOLSTA2
                  kfloref(is6bsn-6) = kfloref(is6bsn-6) + RMKVOLSTA2
                  flow_to_import = flow_to_import + RMKVOLSTA2
                  flow_to_import_hill = flow_to_import_hill +
     $            RMKVOLSTA2
                  VOLWTR(INDXSTA(IS7BSN-6)) = VOLWTR(INDXSTA
     $            (IS7BSN-6)) - RMKVOLSTA2
               ENDIF
            ELSE
               RMKVOLSTA2 = 0.0
            ENDIF
         ELSE
            IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
```

```
     $      (idn_wca_indx(is8bsn-6)) .LE.regstg_wca(iwca3a,2) .AND.
     $      trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca
     $      (iwca2a,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6))
     $      .LE. regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ.
     $      'FALSE' .AND. trig_wca_stg(idn_wca_indx(is6bsn-6)) .LE.
     $      regstg_wca(iwca1,1))) THEN
               IF (KFLO(IFLS2) .GE. 0.) THEN
                  RMKVOLSTA2 = AMIN1(TARGSTA2 ,AMAX1(FLLKS6-KFLO
     $            (IFLS6)-dmnd_hill,0.0) ,amax1(s351mx-kflo(ifls2),
     $            0.0),s6max-kflo(ifls6))
                  RMKVOLSTA2 = AMAX1(RMKVOLSTA2,0.0)
                  KFLO(IFLS2) = KFLO(IFLS2) + RMKVOLSTA2
                  KFLO(IFLS6) = KFLO(IFLS6) + RMKVOLSTA2
c
                  kfloref(is6bsn-6) = kfloref(is6bsn-6) + RMKVOLSTA2
                  flow_to_import = flow_to_import + RMKVOLSTA2
                  flow_to_import_hill = flow_to_import_hill +
     $            RMKVOLSTA2
               ENDIF
            ELSE
               RMKVOLSTA2 = 0.0
            ENDIF
         ENDIF
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF ((makeup_water_restr .EQ. 'TRUE ' .AND. trig_wca_stg
     $   (idn_wca_indx(is8bsn-6)) .LE.regstg_wca(iwca3a,2) .AND.
     $   trig_wca_stg(idn_wca_indx(is7bsn-6)) .LE. regstg_wca(iwca2a
     $   ,1) .AND. trig_wca_stg(idn_wca_indx(is6bsn-6)) .LE.
     $   regstg_wca(iwca1,1)) .OR. (makeup_water_restr .EQ. 'FALSE'
     $   .AND. trig_wca_stg(idn_wca_indx(is6bsn-6)) .LE. regstg_wca
     $   (iwca1,1))) THEN
            IF (istaopt(is5absn-6) .EQ. 1) THEN
               RMKVOLSTA1 = AMIN1(TARGSTA1,VOLWTR(INDXSTA(IS5ABSN-6))
     $         ,AMAX1(FLLKWPB-AMAX1(KFLO(IFLS5A),kflo(iflhg5)),0.0)
     $         ,amax1(s352mx - kflo(iflhg5),0.0),s5amax-kflo(ifls5a))
               RMKVOLSTA1 = AMAX1(RMKVOLSTA1,0.0)
               RCAPS5A = S5AMAX - KFLO(IFLS5A)
               RMKVOLSTA1 = AMIN1(RMKVOLSTA1,RCAPS5A)
c
               kfloref(is5absn-6) = kfloref(is5absn-6) + RMKVOLSTA1
               flow_to_import = flow_to_import + RMKVOLSTA1
               flow_to_import_wpb = flow_to_import_wpb + RMKVOLSTA1
               KFLO(IFLHG5) = KFLO(IFLHG5) + RMKVOLSTA1
               KFLO(IFLS5A) = KFLO(IFLS5A) + RMKVOLSTA1
               KFLO(IFLS5A5) = KFLO(IFLS5A5) + RMKVOLSTA1
               s5a2_so = s5a2_so + RMKVOLSTA1
               VOLWTR(INDXSTA(IS5ABSN-6)) = VOLWTR(INDXSTA(IS5ABSN
     $         -6)) - RMKVOLSTA1
            ELSE
               RMKVOLSTA1 = AMIN1(TARGSTA1,AMAX1 (FLLKWPB-AMAX1(KFLO
     $         (IFLS5A), kflo(iflhg5)),0.0) ,amax1(s352mx - kflo
     $         (iflhg5),0.0),s5amax - kflo(ifls5a))
               RMKVOLSTA1 = AMAX1(RMKVOLSTA1,0.0)
               RCAPS5A = S5AMAX - KFLO(IFLS5A)
               RMKVOLSTA1 = AMIN1(RMKVOLSTA1,RCAPS5A)
c
               kfloref(is5absn-6) = kfloref(is5absn-6) + RMKVOLSTA1
```

```
               flow_to_import = flow_to_import + RMKVOLSTA1
               flow_to_import_wpb = flow_to_import_wpb + RMKVOLSTA1
c
               KFLO(IFLHG5) = KFLO(IFLHG5) + RMKVOLSTA1
               KFLO(IFLS5A) = KFLO(IFLS5A) + RMKVOLSTA1
               KFLO(IFLS5A5) = KFLO(IFLS5A5) + RMKVOLSTA1
               s5a2_so = s5a2_so + RMKVOLSTA1
            ENDIF
         ELSE
            RMKVOLSTA1 = 0.0
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C    COMPUTE TARGET NOT MET FOR TIME STEP
         TARGNOTMT = TARGSTA3 - RMKVOLSTA3 + TARGSTA2 - RMKVOLSTA2
     $   +  TARGSTA1 - RMKVOLSTA1 + TARGSTA4 - RMKVOLSTA4
CBW CBW/
      ELSE
         TARGNOTMT = 0.0
         TARGET = 0.0
         RMKVOLSTA1 = 0.0
         RMKVOLSTA2 = 0.0
         RMKVOLSTA3 = 0.0
         RMKVOLSTA4 = 0.0
      ENDIF
      RETURN
      END
c---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

flw298dso.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: flw298dso.inc,v 1.4 2003/11/17 19:41:28 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/flw298dso.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /flw298dso/ flow_to_sta3_fr_298_nnrc,flow_to_sta2_fr_298d ,
     +add_flow_to_sta34,flow_to_eaa_res_nnr,flow_to_sta1w_fr_298d ,
     +sugh_runoff_over_des_strsta6
```

funcs.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: funcs.F,v 1.24 2005/06/10 13:29:05 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/funcs.F,v $
C   $Name: rel-5-5 $
C
C
C    1 -    REAL FUNCTION s340sch
C    2 -    REAL FUNCTION volenp
C    3 -    REAL FUNCTION vol332
C    4 -    REAL FUNCTION vol332do
C    5 -    REAL FUNCTION vol18c
C    6 -         FUNCTION get_tide
C    7 -         FUNCTION tideinterp
C    8 -    REAL FUNCTION loketvol
C    9 -    REAL FUNCTION loketlitz
C   10 -    REAL FUNCTION loketres
C   11 -    REAL FUNCTION stg2ar
C   12 -    REAL FUNCTION stg2arlitz
C   13 -    REAL FUNCTION stg2arres
C   14 -    REAL FUNCTION stg2mar
C   15 -         FUNCTION julian
C   16 - INTEGER FUNCTION julian_ymd
C   17 - INTEGER FUNCTION idofw
C   18 -    REAL FUNCTION small_reserv_infilt
C   19 -    REAL FUNCTION g596mx
C   20 -    REAL FUNCTION strout
C   21 -    REAL FUNCTION strout2
C   22 -    REAL FUNCTION cgseep
C   23 -    REAL FUNCTION tovflc
C   24 -         FUNCTION monchk
C   25 -         FUNCTION sched
C   26 -         FUNCTION ainterp
C   27 -         FUNCTION schedwt
C   28 -         FUNCTION schedwn
C   29 -         FUNCTION sched1
```

```
C   30 -         FUNCTION ssmwet
C   31 -    REAL FUNCTION lvseepcfs
C   32 -    REAL FUNCTION stg2sto
C   33 -    REAL FUNCTION stg2stolitz
C   34 -    REAL FUNCTION stg2stores
C   35 - INTEGER FUNCTION iag
C   36 - INTEGER FUNCTION nmosdiff
C   37 -    REAL FUNCTION est_etiu
C
C-------------------------------------------------------------------------------

C*******************************************************************************

C   GIVES OPERATION SCHEDULE FOR S340 AND S339 BASED ON GAGE 3A-2

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION s340sch (month, iday)

      DIMENSION numd(12)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      DATE = MONTH + FLOAT(IDAY)/FLOAT(NUMD(MONTH))

      IF(DATE .LE. 2.0) THEN
         S340SCH = -.165 * (DATE - 1.0) + 11.66
      ELSEIF(DATE .LT. 5.645) THEN
         S340SCH = 11.50
      ELSEIF(DATE .LT. 9.0) THEN
         S340SCH = .149 * (DATE - 5.645) + 11.50
      ELSEIF(DATE .LT. 11.0) THEN
         S340SCH = 12.00
      ELSEIF(DATE .LT. 12.0) THEN
         S340SCH = -.25 * (DATE - 11.0) + 12.0
      ELSE
         S340SCH = -.093 * (DATE -12.0) + 11.75
      ENDIF

      RETURN
      END

C*******************************************************************************

C   FINDS DAILY DELIVERY SCHEDULE TO EVERGLADES NATIONAL PARK

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION volenp (month)

      DIMENSION numd(12)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      IF( MONTH .EQ. 1) THEN
         ACREFT = 22000.
      ELSEIF( MONTH .EQ. 2) THEN
```

```
         ACREFT = 9000.
      ELSEIF( MONTH .EQ. 3) THEN
         ACREFT = 4000.
      ELSEIF( MONTH .EQ. 4) THEN
         ACREFT = 1700.
      ELSEIF( MONTH .EQ. 5) THEN
         ACREFT = 1700.
      ELSEIF( MONTH .EQ. 6) THEN
         ACREFT = 5000.
      ELSEIF( MONTH .EQ. 7) THEN
         ACREFT = 7400.
      ELSEIF( MONTH .EQ. 8) THEN
         ACREFT = 12200.
      ELSEIF( MONTH .EQ. 9) THEN
         ACREFT = 39000.
      ELSEIF( MONTH .EQ. 10) THEN
         ACREFT = 67000.
      ELSEIF( MONTH .EQ. 11) THEN
         ACREFT = 59000.
      ELSEIF( MONTH .EQ. 12) THEN
         ACREFT = 32000.
      ENDIF

      VOLENP = ACREFT*43560./(NUMD(MONTH))

      RETURN
      END

C*****************************************************************************

C   FINDS DAILY DELIVERY SCHEDULE FOR TAYLOR SLOUGH

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION vol332 (month)


      DIMENSION numd(12)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      IF(MONTH .EQ. 1) THEN
         VOL332 = 12.
      ELSEIF(MONTH .EQ. 2) THEN
         VOL332 = 6.7
      ELSEIF(MONTH .EQ. 3) THEN
         VOL332 = 3.0
      ELSEIF(MONTH .EQ. 4) THEN
         VOL332 = 3.1
      ELSEIF(MONTH .EQ. 5) THEN
         VOL332 = 6.0
      ELSEIF(MONTH .EQ. 6) THEN
         VOL332 = 112.
      ELSEIF(MONTH .EQ. 7) THEN
         VOL332 = 120.
      ELSEIF(MONTH .EQ. 8) THEN
         VOL332 = 48.
```

```
      ELSEIF(MONTH .EQ. 9) THEN
         VOL332 = 100.
      ELSEIF(MONTH .EQ. 10) THEN
         VOL332 = 126.
      ELSEIF(MONTH .EQ. 11) THEN
         VOL332 = 62.
      ELSEIF(MONTH .EQ. 12) THEN
         VOL332 = 12.
      ENDIF

      RETURN
      END

C*****************************************************************************

C   MINIMUM DELIVERY TO MAINTAIN DESIRABLE DISSOLVED OXYGEN DOWNSTREAM OF S332

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION vol332do (month)

      DIMENSION numd(12)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      IF(MONTH .EQ. 1) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 2) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 3) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 4) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 5) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 6) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 7) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 8) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 9) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 10) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 11) THEN
         VOL332DO = 10.
      ELSEIF(MONTH .EQ. 12) THEN
         VOL332DO = 10.
      ENDIF

      RETURN
      END

C*****************************************************************************

C   FINDS DAILY DELIVERY SCHEDULE TO ENP THROUGH S-18C
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION vol18c (month)

      DIMENSION numd(12)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      IF(MONTH .EQ. 1) THEN
         ACRFT = 1540.
      ELSEIF(MONTH .EQ. 2) THEN
         ACRFT = 630.
      ELSEIF(MONTH .EQ. 3) THEN
         ACRFT = 290.
      ELSEIF(MONTH .EQ. 4) THEN
         ACRFT = 110.
      ELSEIF(MONTH .EQ. 5) THEN
         ACRFT = 110.
      ELSEIF(MONTH .EQ. 6) THEN
         ACRFT = 340.
      ELSEIF(MONTH .EQ. 7) THEN
         ACRFT = 510.
      ELSEIF(MONTH .EQ. 8) THEN
         ACRFT = 860.
      ELSEIF(MONTH .EQ. 9) THEN
         ACRFT = 2690.
      ELSEIF(MONTH .EQ. 10) THEN
         ACRFT = 4630.
      ELSEIF(MONTH .EQ. 11) THEN
         ACRFT = 4060.
      ELSEIF(MONTH .EQ. 12) THEN
         ACRFT = 2230.
      ENDIF

      VOL18C = ACRFT*43560./(NUMD(MONTH))

      RETURN
      END

C*****************************************************************************

C   assign known daily tides at these 6 locations; for more information on these
see memo from
C   N Krishnan and C Gove to C. Neidrauer, LEC Planning Division, Sep. 20, 1993
C   added flamingo to the original 5 stations.

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION get_tide (cell, jday, proj_sea_level_rise)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'

      DIMENSION ycell(7)

      INTEGER cell
```

```
      REAL kwntide(7)

C   at these locations assign the y cell number to ycell and today's knowntide
to kwntide

      ycell(1)=1
      kwntide(1)=yval(jday) + proj_sea_level_rise
      ycell(2)=4
      kwntide(2)=yval1(jday) + proj_sea_level_rise
      ycell(3)=23
      kwntide(3)=yval2(jday) + proj_sea_level_rise
      ycell(4)=31
      kwntide(4)=yval3(jday) + proj_sea_level_rise
      ycell(5)=46
      kwntide(5)=yval4(jday) + proj_sea_level_rise
      ycell(6)=55
      kwntide(6)=yval5(jday) + proj_sea_level_rise
      ycell(7)=71
      kwntide(7)=yval6(jday) + proj_sea_level_rise

C   interpolate in y direction to get current cells tide

      y=cell
      get_tide = ainterp(y,ycell,kwntide,7,0.,0.,0)

      RETURN
      END

C*******************************************************************************

C   linearly interpolates within a real array (y) using a real array (x) as
ordinates. assumes
C   ordinates are in ascending order

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION tideinterp (xval, x, y, n)

      DIMENSION y(n)

      INTEGER x(n)

C   if date is < jan 15 or > dec 15  then interpolate between dec15 and jan15
values and return

      IF (xval .LE. x(1) .OR. xval .GT. x(n)) THEN
         IF (xval .GT. 349) THEN
            xval=xval-349
         ELSE
            xval=xval+16
         ENDIF
         tideinterp = xval/31.0*(y(1) - y(n)) + y(n)
         RETURN
      ENDIF

C   normal interpolation
```

```
      DO i = 1,n - 1
         IF (xval .GE. x(i) .AND. xval .LE. x(i+1)) goto 10
      END DO
   10 CONTINUE
      tideinterp = (xval - x(i))/(x(i+1) - x(i))*(y(i+1) - y(i)) + y(i)

      RETURN
      END

C*****************************************************************************

C     computes daily volume of evapotranspiration from Lake Okeechobee
C     calculates  evapotranspiration from "dry" littoral zone,
C     inundated littoral zone, open water, and sums for lake total.
C     calls REAL functions stg2ar and stg2mar for surface areas of
C     corresponding lake zones.
C
C     variable names:
C     X         pan  evaporation coefficient
C     LOKSA     Lake Okeechobee Surface Area at stage 20 ft NGVD (acres)
C     K         evapotranspiration coefficient
C     MODRYET   Monthly Dry Littoral Evapotranspiration (inches)
C     MOPAN     Monthly Pan Evaporation (inches)
C     AVLMOET   Available Monthly Evapotranpiration (inches)
C     LOKMORN   Monthly Lake Okeechobee Rainfall (inches)
C     DADRYET   Daily Dry Littoral Evapotranspiration (inches)
C     DAPAN     Daily Pan Evaporation (inches)
C     LOKWTRSA  Lake Okeechobee Water Surface Area (acres)
C     DRYSURF   Lake Okeechobee Dry Littoral Surface Area (acres)
C     DRYETVOL  Daily Dry Littoral Evapotranspiration Volume (ac-ft)
C     LOKLITSA  Lake Okeechobee wet Littoral Surface Area (acres)
C     LITETVOL  Daily wet Littoral Evapotranspiration Volume (ac-ft)
C     LOKOPNSA  Lake Okeechobee Open Water Surface Area (acres)
C     OPNETVOL  Daily Open Water Evaporation Volume (ac-ft)
C     LOKETVOL  total Lake Okeechobee Evapotranspiration Volume (ac-ft)
C
C     data sources: X, K - S. F. Shih (1980);
C     LOKMORN - S. Sculley (1986) updated to 1990;
C     MOREFET, DAREFET - 3 gage average: BELLE GLADE (NOAA), CANAL POINT
C     (NOAA), MOORE HAVEN (NOAA)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION loketvol (stg, month, rf_factor_lok)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'et.inc'

      REAL avlmoet, dadryet, darefet, dryetvol, drysurf, k, loksa,
     & lokwtrsa, modryet, opnetvol, rf_factor_lok(12), x

C   coefficient to convert reference et to open  water 1.27*.865= 1.1

      X =1.0
      LOKSA = 466000.0
      K = 1.2
```

```
C   "Dry" Littoral Zone: volumetric calculation of ET

      MODRYET = K * X * MOREFET(MONTH)
      AVLMOET = MIN (MODRYET,LOKMORN(MONTH)*rf_factor_lok(MONTH))
      DADRYET = DAREFET * AVLMOET / MOREFET(MONTH)

      LOKWTRSA = STG2AR(STG)
      DRYSURF = LOKSA - LOKWTRSA
      DRYETVOL = DADRYET * DRYSURF / 12.0

C   Inundated Littoral Zone: volumetric calculation of ET

      LOKLITSA = STG2MAR(STG)
      LITETVOL = LOKLITSA * X * K * DAREFET / 12.0

C   Lake Open Water Zone: volumetric calculation of ET

      LOKOPNSA = LOKWTRSA - LOKLITSA
      OPNETVOL = LOKOPNSA * X * DAREFET / 12.0

C   Total Lake Zone(s): volumetric calculation of ET

      LOKETVOL = DRYETVOL + LITETVOL + OPNETVOL

      RETURN
      END

C*****************************************************************************

C     computes daily volume of evapotranspiration from Lake Okeechobee
C     calculates  evapotranspiration from "dry" littoral zone,
C     inundated littoral zone, open water, and sums for lake total.
C     calls REAL functions stg2ar and stg2mar for surface areas of
C     corresponding lake zones.
C
C     variable names:
C     X         pan  evaporation coefficient
C     LOKSA     Lake Okeechobee Surface Area at stage 20 ft NGVD (acres)
C     K         evapotranspiration coefficient
C     MODRYET   Monthly Dry Littoral Evapotranspiration (inches)
C     MOPAN     Monthly Pan Evaporation (inches)
C     AVLMOET   Available Monthly Evapotranpiration (inches)
C     LOKMORN   Monthly Lake Okeechobee Rainfall (inches)
C     DADRYET   Daily Dry Littoral Evapotranspiration (inches)
C     DAPAN     Daily Pan Evaporation (inches)
C     LOKWTRSA  Lake Okeechobee Water Surface Area (acres)
C     DRYSURF   Lake Okeechobee Dry Littoral Surface Area (acres)
C     DRYETVOL  Daily Dry Littoral Evapotranspiration Volume (ac-ft)
C     LOKLITSA  Lake Okeechobee wet Littoral Surface Area (acres)
C     LITETVOL  Daily wet Littoral Evapotranspiration Volume (ac-ft)
C     LOKOPNSA  Lake Okeechobee Open Water Surface Area (acres)
C     OPNETVOL  Daily Open Water Evaporation Volume (ac-ft)
C     LOKETVOL  total Lake Okeechobee Evapotranspiration Volume (ac-ft)
C
C     data sources: X, K - S. F. Shih (1980);
C     LOKMORN - S. Sculley (1986) updated to 1990;
```

```
C     MOREFET, DAREFET - 3 gage average: BELLE GLADE (NOAA), CANAL POINT
C     (NOAA), MOORE HAVEN (NOAA)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION loketlitz (stg, month, rf_factor_lok)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'et.inc'

      REAL avlmoet, dadryet, darefet, dryetvol, drysurf, k, loksa,
     & lokwtrsa, modryet, opnetvol, rf_factor_lok(12), x

C   coefficient to convert reference et to open  water 1.27*.865= 1.1

      X = 1.0
      LOKSA = 279600.0
      K = 1.2

C   "Dry" Littoral Zone: volumetric calculation of ET

      MODRYET = K * X * MOREFET(MONTH)
      AVLMOET = MIN (MODRYET,LOKMORN(MONTH)*rf_factor_lok(month))
      DADRYET = DAREFET * AVLMOET / MOREFET(MONTH)
C
      LOKWTRSA = STG2ARLITZ(STG)
      DRYSURF = LOKSA - LOKWTRSA
      DRYETVOL = DADRYET * DRYSURF / 12.0

C   Inundated Littoral Zone: volumetric calculation of ET

      LOKLITSA = STG2MAR(STG)
      LITETVOL = LOKLITSA * X * K * DAREFET / 12.0

C   Lake Open Water Zone: volumetric calculation of ET

      LOKOPNSA = LOKWTRSA - LOKLITSA
      OPNETVOL = LOKOPNSA * X * DAREFET / 12.0

C   Total Lake Zone(s): volumetric calculation of ET

      LOKETLITZ = DRYETVOL + LITETVOL + OPNETVOL

      RETURN
      END

C*****************************************************************************

C     computes daily volume of evapotranspiration from Lake Okeechobee reservoir

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION loketres (stg, month, rf_factor_lok)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'lakwca.inc'
```

```
      INCLUDE 'et.inc'

      REAL avlmoet, dadryet, darefet, dryetvol, drysurf, k, loksa,
     & lokwtrsa, modryet, opnetvol, rf_factor_lok(12), x


C   coefficient to convert reference et to open  water 1.27*.865= 1.1

      X =1.0
      LOKSA = 186400.0
      K = 1.2

C   Lake Open Water Zone: volumetric calculation of ET

      LOKWTRSA = STG2ARRES(STG)

C   "Dry" Littoral Zone: volumetric calculation of ET

      MODRYET = K * X * MOREFET(MONTH)
      AVLMOET = MIN (MODRYET,LOKMORN(MONTH)*rf_factor_lok(month))
      DADRYET = DAREFET * AVLMOET / MOREFET(MONTH)

      DRYSURF = LOKSA - LOKWTRSA
      DRYETVOL = DADRYET * DRYSURF / 12.0
      OPNETVOL = LOKWTRSA * X * DAREFET / 12.0

C   Total Lake Zone(s): volumetric calculation of ET

      LOKETRES = OPNETVOL + DRYETVOL

      RETURN
      END

C*****************************************************************************

C     computes the water surface area of Lake Okeechobee (acres) from
C     a given stage (feet NGVD).  REAL function stg2ar calls REAL
C     function ainterp to linearly interpolate between breakpoints

C      data source: Lake Okeechobee Area-Capacity Curve; U.S. Army
C      Corps of Engineers; Jacksonville, Fla., District; May 1962 --
C      Area Capacity Lake Okeechobee (table); U.S. Army Corps of
C      Engineers; Jacksonville, Fla., District; 2 July 1962
C
C    Lake Okeechobee: calculation of Surface Area from stage

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2ar (stg)

      REAL area(33), stage(33)

      DATA area / 0.0,75000.0,125000.0,135000.0,190000.,250000.0,
     & 260000.0, 270000.0, 284000.0, 299000.0, 306800.0, 316000.,
     & 326000.0, 339000.0, 355000.0, 371000.0, 383600.0, 395000.0,
     & 417200.0, 436000.0, 443900.0, 448200.0, 450200.0, 452200.0,
     & 454100.0, 455800.0, 457300.0, 459000.0, 460700.0, 462400.0,
```

```
     & 464000.0, 466000.0, 466000. 0/
      DATA stage /0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 9.5,
     & 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5, 15.0,
     & 15.5, 16.0, 16.5, 17.0, 17.5, 18.0, 18.5, 19.0, 19.5, 20.0, 50.0/

      STG2AR = AINTERP(STG,STAGE,AREA,33,0.,0.,0)

      RETURN
      END

C*****************************************************************************

C     computes the water surface area of western Lake Okeechobee (acres)
C     from a given stage (feet NGVD).  REAL function stg2ar calls REAL
C     function ainterp to linearly interpolate between breakpoints

C      data source: Lake Okeechobee Area-Capacity Curve; U.S. Army
C      Corps of Engineers; Jacksonville, Fla., District; May 1962 --
C      Area Capacity Lake Okeechobee (table); U.S. Army Corps of
C      Engineers; Jacksonville, Fla., District; 2 July 1962

C    Lake Okeechobee: calculation of Surface Area from stage

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2arlitz (stg)

      REAL area(33), stage(33)

      DATA area / 0.0,10.0,17.0,6559.0,42541.0,78079.0,84002.0, 95303.0,
     & 111124.0, 125572.0, 133085.0, 141947.0, 151373.0, 163626.0,
     & 178708.0, 193789.0, 205740.0, 216552.0, 237607.0, 255437.0,
     & 266340.0, 268920.0, 270120.0, 271320.0, 272460.0, 273480.0,
     & 274380.0, 275400.0, 276420.0, 277440.0, 278400.0, 279600.0,
     & 279600. 0/
      DATA stage /0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 9.5,
     & 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5, 15.0,
     & 15.5, 16.0, 16.5, 17.0, 17.5, 18.0, 18.5, 19.0, 19.5, 20.0, 50.0/

      STG2ARLITZ = AINTERP(STG,STAGE,AREA,33,0.,0.,0)

      RETURN
      END

C*****************************************************************************

C     computes the water surface area of eastern Lake Okeechobee (acres)
C     from a given stage (feet NGVD).  REAL function stg2ar calls REAL
C     function ainterp to linearly interpolate between breakpoints

C      data source: Lake Okeechobee Area-Capacity Curve; U.S. Army
C      Corps of Engineers; Jacksonville, Fla., District; May 1962 --
C      Area Capacity Lake Okeechobee (table); U.S. Army Corps of
C      Engineers; Jacksonville, Fla., District; 2 July 1962

C    Lake Okeechobee: calculation of Surface Area from stage
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2arres (stg)

      REAL area(33), stage(33)

      DATA area / 0.0,74990.0,124983.0,128441.0,147459.0,171921.0,
     & 175998.0,174697.0, 172876.0, 173428.0, 173715.0, 174053.0,
     & 174627.0, 175374.0, 176292.0, 177211.0, 177860.0, 178448.0,
     & 179593.0, 180563.0, 177560.0, 179280.0, 180080.0, 180880.0,
     & 181640.0, 182320.0, 182920.0, 183600.0, 184280.0, 184960.0,
     & 185600.0, 186400.0, 186400. 0/
      DATA stage /0.0, 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 9.5,
     & 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5, 15.0,
     & 15.5, 16.0, 16.5, 17.0, 17.5, 18.0, 18.5, 19.0, 19.5, 20.0, 50.0/

      STG2ARRES = AINTERP(STG,STAGE,AREA,33,0.,0.,0)

      RETURN
      END

C*****************************************************************************

C     computes the inundated littoral zone surface area of Lake
C     Okeechobee (acres) from a given lake stage (feet NGVD).
C     REAL function stg2ar calls REAL function ainterp to linearly
C     interpolate between breakpoints

C      data source: Hosung Ahn's assumption that Lake Okeechobee
C      maximum open water surface area occurs at 11.5 ft NGVD and
C      that increased surface area due to rise in stage is marsh.
C      Thus the littoral zone begins at 11.5 ft NGVD and is computed
C      as 355000.0 acres less than Lake Okeechobee surface area.
C      Lake Okeechobee Area-Capacity Curve; U.S. Army
C      Corps of Engineers; Jacksonville, Fla., District; May 1962 --
C      Area Capacity Lake Okeechobee (table); U.S. Army Corps of
C      Engineers; Jacksonville, Fla., District; 2 July 1962

C   Inundated Littoral Zone: calculation of Surface Area

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2mar (stg)

      REAL area(20), stage(20)

      DATA area /0.00, 0.00, 1038.29, 14536.00, 18964.43, 45535.00,
     & 47014.58, 49639.44, 68013.42, 71260.75, 77755.39, 79857.44,
     & 84061.54, 85196.32, 88600.67, 89298.58, 92090.25, 92165.70,
     & 93222.00, 93222.00/
      DATA stage / 0., 11.0, 11.1, 12.4, 12.5, 13.1, 13.2, 13.3, 14.0,
     & 14.1, 14.3, 14.4, 14.6, 14.7, 15.0, 15.1, 15.5, 15.6, 17.0, 50.0/

      STG2MAR = AINTERP(STG,STAGE,AREA,20,0.,0.,0)

      RETURN
      END
```

```
C*****************************************************************************

C   computes a julian day number

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION julian (month, iday, numdays)

      INTEGER numdays(12)

      JULIAN = 0
      DO 10 I = 1,MONTH-1
         JULIAN = JULIAN + NUMDAYS(I)
   10 CONTINUE
      JULIAN = JULIAN + IDAY

      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      INTEGER FUNCTION julian_ymd (igregyear, igregmonth, igregday)

      IF (igregmonth .GT. 2) THEN
         igregmonth = igregmonth - 3
      ELSE
         igregmonth = igregmonth + 9
         igregyear = igregyear - 1
      ENDIF
      ic = igregyear/100
      iya = igregyear - (100*ic)
      julian_ymd = (146097*ic)/ 4+ (1461*iya)/ 4+ (153*igregmonth + 2)/
     & 5 + igregday + 1721119

      RETURN
      END

C*****************************************************************************

C   This function determines the day of week a user-specified date falls into.
It uses the
C   Julian Day concept and the mod() function to do the determination. lmb
2/2001
C   Sunday = 1, . . ., Saturday = 7

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      INTEGER FUNCTION idofw (mm, id, iyyy)

      PARAMETER (igreg = 15 + 31*(10 + 12*1582))

      if (iyyy.eq.0) then
        print *,'Error in function idofw:'
        print *,'There is no Year Zero.'
```

```
        stop
      endif
      if (iyyy.lt.0) iyyy = iyyy + 1
      if (mm.gt.2) then
         jy = iyyy
         jm = mm + 1
      else
         jy = iyyy - 1
         jm = mm + 13
      endif
      jday = int(365.25*jy) + int(30.6001*jm) + id + 1720995
      if (id + 31*(mm + 12*iyyy).ge.igreg) then
         ja = int(0.01*jy)
         jday = jday + 2 - ja + int(0.25*ja)
      endif

      nnn = mod(jday+1,7)
      idofw = nnn + 1

      RETURN
      END

C*****************************************************************************

C   function to calculate infiltration to water table from surface water in
small reservoir

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION small_reserv_infilt (iresnum, ells, pond,
     & pond_in_res, h, ibsn, sfactor, s, node, ratio_inf_sw, sinf, dt)

      INCLUDE 'wmm_par.inc'

      DIMENSION ells(max_ncells), h(max_ncells), ibsn(max_ncells),
     & pond(max_ncells), pond_in_res(max_ncells), s(max_ncells),
     & sfactor(max_n_res), sinf(max_ncells)

      small_reserv_infilt = 0.0
      avail_gw_dpth = (ells(node) - h(node)) * s(node)
      avail_gw_dpth = amin1(avail_gw_dpth,sinf(node)*dt)
      avail_surfw_dpth = pond(node)*(1.0-sfactor(iresnum))
     & + pond_in_res(node)*sfactor(iresnum)
      IF (avail_surfw_dpth .gt. 0.000001) THEN
         ratio_inf_sw = amin1(avail_gw_dpth/avail_surfw_dpth,1.0)
         small_reserv_infilt =ratio_inf_sw * pond_in_res(node)
      ENDIF

      RETURN
      END

C*****************************************************************************

C   function to find regulation stage from given regulation schedule at gage 596
in east everglades

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
      REAL FUNCTION g596mx (month, iday)

      DIMENSION NUMD(12)

      DATA NUMD/31,28,31,30,31,30,31,31,30,31,30,31/

      DATE = MONTH + FLOAT(IDAY)/FLOAT(NUMD(MONTH))

      IF( DATE .GT. 6.5 .AND. DATE .LT. 10.) THEN
         G596MX = 6.5
      ELSEIF( DATE .GE. 5. AND. DATE. LE. 6.5 ) THEN
         G596MX = 1. 6667* (DATE - 5.0) + 4.0
      ELSEIF( DATE .GE. 10. ) THEN
         G596MX = -. 357143* ( DATE - 10.) + 6.5
      ELSE
         G596MX = -. 357143* ( DATE - 1.0) + 5.43
      ENDIF

      RETURN
      END


C*****************************************************************************
C     THIS FUNCTION CALCULATES DOWNSTREAM DISCHARGE AT A SPECIFIED
C     STRUCTURE FOR SPECIFIED HEADWATER AND SIMULATED TAILWATER. THIS
C     FUNCTION IS USED TO HELP DETERMINE THE MAXIMUM (LIMIT) DISCHARGE
C     THAT COULD BE DELIVERED THROUGH THE STRUCTURE WITHOUT VIOLATING
C     TAILWATER CONSTRAINTS.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION strout (stagem, m, n, ind1, inflowopt, crmin,
     & volwtr, prevqu, rain, iter, tot_pump, jday, tempf_volwtr,
     & tempf_weircff, iactual_flow_sim)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'cnlneeds.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'dualops.inc'

      DIMENSION capacflf(max_n_seas), capacwsf(max_n_seas),
     & crmin(max_ncnls,max_n_seas), prevqu(max_ncnls),
     & qinflown(max_ncnls), rain(max_ncells), str_cap(5),
     & tempf_volwtr(max_n_res), tempf_weircff(max_n_res,max_nostr_res),
```

```
     & volwtr(max_n_res)

      LOGICAL langel_mxx, langelx, ldishgf, logdisgf

      CHARACTER char_struc_name*6
      DOUBLE PRECISION crmin,stagem,stg_rmx,STGDN,rmin_stg_fc,tw
     $,stagediff,stagedff,rdn_stg_limf,rminstgf,rminstgf_dn

      IFLO = ISTR(N,IND1)
      char_struc_name = flnm(iflo)
      IDNCLIDX = IDNCNL(N,IND1)
      IF (ITER .EQ. 0) THEN
         QINFLOWN(N) = PREVQU(N)
      ENDIF
C
      IDSHGF = 0
      IF (NDSCLIM(N,IND1) .GT. 1) THEN
         DO 10 I = 1,NDSCLIM(N,IND1)
            IF (CHDEP(ICNUM(N,IND1,I)) .GE. CNLSTG(N,IND1,I)) THEN
               IDSHGF = IDSHGF + 1
            ENDIF
   10    CONTINUE
      ENDIF
      LDISHGF = IDSHGF .EQ. NDSCLIM(N,IND1)
C
      IF (NDSCLIM(N,IND1) .GT. 1) THEN
         IF (FLNM(IFLO) .EQ. strname_spec(13)) THEN
            LOGDISGF = LDISHGF
         ELSE
            LOGDISGF = .NOT. LDISHGF
         ENDIF
      ELSE
         LOGDISGF = .TRUE.
      ENDIF
C
C
      IF (IRCEIVE(N,IND1) .EQ. 1) THEN
         IF (IDNCLIDX .LE. NCH) THEN
            STGDIFF_CANAL_TW = FLOAT(NODCR(IDNCLIDX)-NODEPOS_SD(N ,IND1)
     &      ) /FLOAT(NODCR(IDNCLIDX)) * HDC(IDNCLIDX,iseas)
            TW= CHDEP(IDNCLIDX) + STGDIFF_CANAL_TW
         ENDIF
         IF (FLNM(IFLO) .EQ. strname_spec(37).OR.FLNM(IFLO) .EQ.
     &   strname_spec(21)) TW = STAGELO
         IF (icell_no_sd(n,ind1) .gt. 0) THEN
           IF (lutyp(icell_no_sd(n,ind1)) .eq. 0) then
              twdph=get_tide(ycnlnode(n,1),jday,
     &         proj_sea_level_rise)
              tw = 1.025*(twdph + 2.0) - 2.0
           ENDIF
         ENDIF
C
      ELSEIF (IRCEIVE(N,IND1) .EQ. 2 .OR. IRCEIVE(N,IND1) .EQ. 3) THEN

         TW = ELLS(NODRCV(N,IND1,1)) + POND(NODRCV(N,IND1,1))
      ENDIF
C
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
      IF (NCOEFF(N,IND1) .EQ. 1) THEN
         IMGT = 1
      ELSE
         IMGT = ISEAS
      ENDIF
      STROUT = 0.0
C
      DO JJ = 1,NCOEFF(N,IND1)
         CAPACFLF(JJ) = 0.0
         CAPACWSF(JJ) = 0.0
      END DO
C
      idiff_trigc_upstrmcx = 0
      itrg_indxm = 1
      DO isfc_ops = 1,nsfc_w_chg_trg
         IF (n.eq.icnl_no_upstrm_trg_ops(isfc_ops).and.ind1.eq.
     &   istruc_no_chg_trg_ops(isfc_ops)) THEN
            IF (istruc_no_alt_trg_ops(isfc_ops) .gt. 0) THEN
               IF (nint(kflo(istruc_no_alt_trg_ops(isfc_ops))).eq. 0)
     &         THEN
                  itrg_indxm = 2
               ELSE
                  itrg_indxm = 1
               ENDIF
            ELSE
              itrg_indxm = 1
            ENDIF
            IF (node_pos_trg_ops(isfc_ops,itrg_indxm) .gt. 0) THEN
              if (nodcr(icanl_no_trg_ops(isfc_ops,itrg_indxm)).gt.1)
     &        then
                canl_stg_diff_frm_dnstrm = float(nodcr
     &          (icanl_no_trg_ops(isfc_ops,itrg_indxm))-node_pos_trg_ops
     &          (isfc_ops,itrg_indxm))/float(nodcr(icanl_no_trg_ops
     &          (isfc_ops,itrg_indxm))-1)*hdc(icanl_no_trg_ops(isfc_ops,
     &          itrg_indxm),iseas)
              else
                canl_stg_diff_frm_dnstrm = 0.0
              endif
              stg_trig_fc_ops(n,ind1) = chdep(icanl_no_trg_ops
     &        (isfc_ops,itrg_indxm)) + canl_stg_diff_frm_dnstrm
              IF (icnl_no_upstrm_trg_ops(isfc_ops) .ne.
     &        icanl_no_trg_ops(isfc_ops,itrg_indxm))
     &        idiff_trigc_upstrmcx = 1
            ELSE
              stg_trig_fc_ops(n,ind1) =h(igrid_loc_trg_ops(isfc_ops
     &        ,itrg_indxm))+pond(igrid_loc_trg_ops(isfc_ops,itrg_indxm))
              idiff_trigc_upstrmcx = 1
            ENDIF
         ENDIF
      ENDDO

      DO 20 NS = 1,NSUBSTR(N,IND1)
C
         IF (IRATING(N,IND1,NS,IMGT) .EQ. 1) THEN
            IF (FLNM(IFLO) .EQ. strname_spec(37)) THEN
```

```
            DELTAH = STAGEM + HDC(N,iseas) - TW
          ELSE
            DELTAH = STAGEM - TW
          ENDIF
          IF (DELTAH .GT. RMINHD(N,IND1)) THEN
             CAPAC =DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n,ind1,
     &       month) *(DELTAH - RMINHD(N,IND1))** PWR(N,IND1,NS,IMGT)
          ELSE
             CAPAC = 0.0
          ENDIF
          IF (ides_cap_limit(n,ind1) .EQ. 1) THEN
             CAPAC = AMIN1(CAPAC,STRDCAP(N,IND1,NS,IMGT))
          ENDIF
C
          IF (stagem.gt.rmxstg(n,ind1,iseas).or.lgrav(n,ind1,ns)) THEN
             IF (.not. lgrav(n,ind1,ns)) THEN
                IF (stagem.gt.rmxstg(n,ind1,iseas)+0.2) THEN
                   capacfldc0=capac
                ELSE
                   capacfldc0=5.0*capac*(stagem-
     &             rmxstg(n,ind1,iseas))
                END IF
             ELSE
               capacfldc0 = capac
             ENDIF
          ELSE
            capacfldc0 = 0.0
          ENDIF
          capac = capacfldc0
C
       ELSEIF (IRATING(N,IND1,NS,IMGT) .EQ. 2) THEN
          IF (iactual_flow_sim .eq. 0) THEN
             IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS).OR.LPUMP(N,IND1
     &       ,NS)) THEN
                IF (.NOT. LPUMP(N,IND1,NS)) THEN
                   IF (idiff_trigc_upstrmcx .ne. 0) then
                     stg_trig_f_volf = max(stg_trig_fc_ops(n,ind1)
     &               ,stagem)
                   ELSE
                     stg_trig_f_volf = stagem
                   ENDIF
                   IF (stg_trig_f_volf .GT. RMXSTG(N,IND1,ISEAS)
     &             +dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                      CAPAC = DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n
     &                ,ind1,month)
                   ELSE
                      fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                      CAPAC = fmultipl * DISCOEF(N,IND1,NS,IMGT) *
     &                frac_struc_cap(n,ind1,month)*(stg_trig_f_volf -
     &                RMXSTG(N,IND1,ISEAS))
                   ENDIF
                ELSE
                   IF (idiff_trigc_upstrmcx .ne. 0) then
                     stg_trig_f_volf = max(stg_trig_fc_ops(n,ind1)
     &               ,stagem)
                   ELSE
                     stg_trig_f_volf = stagem
```

```
                 ENDIF
                 RMINSTG = RMNSTGS(N,IND1,ISEAS)
                 IF (stg_trig_f_volf .GT. RMINSTG +
&                dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    CAPAC=DISCOEF(N,IND1,NS,IMGT)
&                   * frac_struc_cap(n,ind1,month)
                 ELSEIF(stg_trig_f_volf .GT. RMINSTG) THEN
                    fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                    CAPAC=fmultipl*DISCOEF(N,IND1,NS,IMGT)*
&                   frac_struc_cap(n,ind1,month)*(stg_trig_f_volf
&                   -RMINSTG)
                 ELSE
                    CAPAC=0.0
                 ENDIF
              ENDIF
              IF (LPUMP(N,IND1,NS)) THEN
                 if (imatch_struc_loc(n,ind1) .ne. 0) then
                   rminstgf = rmin_stg_avol(n,ind1)
                 else
                   rminstgf = rmnstgs(n,ind1,iseas)
                 endif
                 IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                  IF (stagem .GT. rminstgf +
&                 dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    AVVOLQ=DISCOEF(N,IND1,NS,IMGT)
&                   * frac_struc_cap(n,ind1,month)
                  ELSEIF(stagem .GT. rminstgf) THEN
                    fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                    AVVOLQ=fmultipl*DISCOEF(N,IND1,NS,IMGT)*
&                   frac_struc_cap(n,ind1,month)*(stagem
&                   -rminstgf)
                  ELSE
                    AVVOLQ=0.0
                  ENDIF
                  IF (nodcr(n) .gt. 1) then
                   stgdiff_canal_f_dns=float(nodcr(n)-nodepos_su(n
&                  ,ind1))/float(nodcr(n)-1)*hdc(n,iseas)
                  ELSE
                   stgdiff_canal_f_dns = 0.0
                  ENDIF
                  rminstgf_dn = rminstgf - stgdiff_canal_f_dns
                  chg_stor_lim = CGSEEP(n,rminstgf_dn,0,0.)-CGSEEP
$                 (n,stagem,0,0.) +TOVFLC(n,rminstgf_dn,0,0.,iter,2)
$                 - TOVFLC(n,stagem,0,0.,iter,2)
$                 + (stagem-rminstgf)*cl(n)*width(n)
                 ELSE
                  IF (nodcr(n) .gt. 1) then
                   stgdiff_canal_f_dns=float(nodcr(n)-nodepos_su(n
&                  ,ind1))/float(nodcr(n)-1)*hdc(n,iseas)
                  ELSE
                   stgdiff_canal_f_dns = 0.0
                  ENDIF
                  IF (stagem .GT. rminstgf +
&                 dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    AVVOLQ_C=DISCOEF(N,IND1,NS,IMGT)
&                   * frac_struc_cap(n,ind1,month)
                  ELSEIF(stagem .GT. rminstgf) THEN
```

```
                        fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                        AVVOLQ_C=fmultipl*DISCOEF(N,IND1,NS,IMGT)*
     &                  frac_struc_cap(n,ind1,month)*(stagem
     &                  -rminstgf)
                      ELSE
                        AVVOLQ_C=0.0
                      ENDIF
                      rminstgf_dn = rminstgf - stgdiff_canal_f_dns
                      AVVOLQ=(stagem-rminstgf)*cl(n)*width(n)
     &                + cgseep(n,rminstgf_dn,0,0.)+TOVFLC(n,rminstgf_dn
     &                ,0,0.,iter,2)+QINFLOWN(N)
                      chg_stor_lim = 0.0
                      AVVOLQ=AMAX1(AVVOLQ,0.0)/86400.
                      AVVOLQ = AMAX1(AVVOLQ_C,AVVOLQ)
                    ENDIF
c                   if (flnm(iflo).eq.'S9A ') print *,STAGEM,capac
c    &               ,AVVOLQ,QINFLOWN(N)/86400.,rminstgf,'  S9AVOL0'
                    CAPAC = AMIN1(AVVOLQ,CAPAC) + chg_stor_lim/86400.
                    CAPAC = MAX(CAPAC,0.0)
                  ELSE
                    if (imatch_struc_loc(n,ind1) .ne. 0) then
                      rminstgf = rmin_stg_avol(n,ind1)
                    else
                      rminstgf = rmnstgs(n,ind1,iseas)
                    endif
                    IF (iopt_for_interdep_on_off(n,ind1).eq.0) THEN
                      IF (stagem .GT. rminstgf +
     &                dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                        AVVOLQ=DISCOEF(N,IND1,NS,IMGT)
     &                  * frac_struc_cap(n,ind1,month)
                      ELSEIF(stagem .GT. rminstgf) THEN
                        fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                        AVVOLQ=fmultipl*DISCOEF(N,IND1,NS,IMGT)*
     &                  frac_struc_cap(n,ind1,month)*(stagem
     &                  -rminstgf)
                      ELSE
                        AVVOLQ=0.0
                      ENDIF
                      IF (nodcr(n) .gt. 1) then
                       stgdiff_canal_f_dns=float(nodcr(n)-nodepos_su(n
     &                 ,ind1))/float(nodcr(n)-1)*hdc(n,iseas)
                      ELSE
                       stgdiff_canal_f_dns=0.0
                      ENDIF
                      rminstgf_dn = rminstgf - stgdiff_canal_f_dns
                      chg_stor_lim = CGSEEP(n,rminstgf_dn,0,0.)-CGSEEP
     $                (n,stagem,0,0.) +TOVFLC(n,rminstgf_dn,0,0.,iter,2)
     $                - TOVFLC(n,stagem,0,0.,iter,2)
     $                + (stagem-rminstgf)*cl(n)*width(n)
                    ELSE
                      IF (nodcr(n) .gt. 1) then
                       stgdiff_canal_f_dns=float(nodcr(n)-nodepos_su(n
     &                 ,ind1))/float(nodcr(n)-1)*hdc(n,iseas)
                      ELSE
                       stgdiff_canal_f_dns = 0.0
                      ENDIF
                      IF (stagem .GT. rminstgf +
```

```
&                 dpth_above_mn_cap_red(n,ind1,iseas)) THEN
                    AVVOLQ_C=DISCOEF(N,IND1,NS,IMGT)
&                   * frac_struc_cap(n,ind1,month)
                  ELSEIF(stagem .GT. rminstgf) THEN
                    fmultipl=1.0/dpth_above_mn_cap_red(n,ind1,iseas)
                    AVVOLQ_C=fmultipl*DISCOEF(N,IND1,NS,IMGT)*
&                   frac_struc_cap(n,ind1,month)*(stagem
&                   -rminstgf)
                  ELSE
                    AVVOLQ_C=0.0
                  ENDIF
                  rminstgf_dn = rminstgf - stgdiff_canal_f_dns
                  AVVOLQ=(stagem-rminstgf)*cl(n)*width(n)
&                 + cgseep(n,rminstgf_dn,0,0.)+TOVFLC(n,rminstgf_dn
&                 ,0,0.,iter,2)+QINFLOWN(N)
                  chg_stor_lim = 0.0
                  AVVOLQ=AMAX1(AVVOLQ,0.0)/86400.
                  AVVOLQ=AMAX1(AVVOLQ_C,AVVOLQ)
                 ENDIF
                 CAPAC = AMIN1(AVVOLQ,CAPAC) + chg_stor_lim/86400.
                 CAPAC = MAX(CAPAC,0.0)
               ENDIF
               TOT_PUMP = TOT_PUMP + CAPAC
            ELSE
              AVVOLQ = 0.0
              CAPAC  = 0.0
            ENDIF
            IF (imatch_struc_loc(n,ind1) .ne. 0 .and. irating(n,ind1
$           ,ns,imgt) .eq. 2 .and. iopt_for_interdep_on_off(n,ind1)
$           .eq.1) THEN
              if (i_counter_mx(imatch_struc_loc(n,ind1)) .eq. 0) THEN
                 rmax_dschg_c = discoef(n,ind1,1,1)
                 isbsmx = 1
                 IF (nsubstr(n,ind1) .gt. 1) THEN
                   DO nss = 2,nsubstr(n,ind1)
                     IF (nint(discoef(n,i,nss,1)) .gt. rmax_dschg_c)
$                    rmax_dschg_c = discoef(n,i,nss,1)
                     isbsmx = nss
                   ENDDO
                 ENDIF
                 insmx = ind1
                 DO i = ind1+1,nstrctr(n)
                   IF (imatch_struc_loc(n,i).eq.imatch_struc_loc(n
$                  ,ind1)) THEN
                     i_counter_mx(imatch_struc_loc(n,i))
$                    = i_counter_mx(imatch_struc_loc(n,i)) + 1
                     DO nss = 1,nsubstr(n,i)
                        IF (nint(discoef(n,i,nss,1)).gt.
$                       nint(rmax_dschg_c)) THEN
                          rmax_dschg_c = discoef(n,i,nss,1)
                          insmx = i
                          isbsmx = nss
                        ENDIF
                     ENDDO
                   ENDIF
                ENDDO
              endif
```

```
                  IF (flnm(iflo).eq. flnm(istr(n,insmx))) THEN
                    dishg_lim_struc = 0.0
                    DO i=1,nstrctr(n)
                      IF (imatch_struc_loc(n,i).eq.imatch_struc_loc(n
     $                ,insmx)) THEN
                        IF (i .ne. insmx) then
                          dishg_lim_struc = discoef(n,i,1,1)
                          insmn = i
                        ENDIF
                      ENDIF
                    ENDDO
                    IF (capac.lt.dishg_lim_struc
     $              .and. stg_trig_fc_ops(n,ind1).gt.rmnstgs(n,insmn
     $              ,iseas)) capac  = 0.0
c                   IF (flnm(iflo).eq.'S9  '.and.nint(outf_fc_thru_s381)
c    $              .eq.0) capac  = 0.0
                  ENDIF
                  if (flnm(iflo).eq.'S9  ') outf_s9_l = capac
                ENDIF
c               IF (flnm(iflo).eq. 'S9  ') THEN
c                  ins9a = 0
c                  dishg_lim_s9a = 0.0
c                  DO i=1,nstrctr(n)
c                  IF (tmpname(n,i).eq. 'S9A   ') THEN
c                    ins9a = i
c                    dishg_lim_s9a = discoef(n,ins9a,1,1)
c                  ENDIF
c                  ENDDO
c               print *,stagem,capac,dishg_lim_s9a,rmnstgs(n,ins9a,iseas)
c    $          ,stg_trig_fc_ops(n,ind1),LPUMP(N,IND1,NS),'  S9LIMB'
c                  IF (capac.lt.dishg_lim_struc
c    $            .and.stg_trig_fc_ops(n,ind1).gt.rmnstgs(n,ins9a,iseas))
c    $            capac = 0.0
c               print *,stagem,capac,dishg_lim_s9a,rmnstgs(n,ins9a,iseas)
c    $          ,stg_trig_fc_ops(n,ind1),LPUMP(N,IND1,NS),'  S9LIM'
c                outf_s9_l = capac
c               ENDIF
c               print *,stagem,rmxstg(N,IND1,NS),RMNSTGS(N,IND1,ISEAS)
c    +          ,AVVOLQ,CAPAC,flnm(iflo),LPUMP(N,IND1,NS)
            ENDIF
         ELSEIF (IRATING(N,IND1,NS,IMGT) .EQ. 3) THEN
            IF (DQU(IDNCLIDX) .GT. 0.0) THEN
               TW = CRMIN(IDNCLIDX,iseas)
            ELSE
               TW = RMNSTGS(N,IND1,ISEAS)
            ENDIF
C
            IF (STAGEM .GT. TW) THEN
               CAPAC =DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n,ind1,
     &         month) * (STAGEM - TW) ** PWR(N,IND1,NS,IMGT)
            ELSE
               CAPAC = 0.0
            ENDIF
         ENDIF
              TVOLMXDN = 0.0
              IF (NDSCLIM(N,IND1) .GT. 0) THEN
                DO NC = 1,NDSCLIM(N,IND1)
```

```
            if (nodcr(icnum(n,ind1,nc)).gt.1) then
              stagediff = float(nodcr(icnum(n,ind1,nc))-nodepos_sd
&              (n,ind1))/float(nodcr(icnum(n,ind1,nc))-1)
&              * hdc(icnum(n,ind1,nc),iseas)
            else
              stagediff = 0.0
            endif
            TDSCHGDN = 0.0
           IF (INTO(ICNUM(N,IND1,NC),1) .GE. 0) THEN
            itide_down = 0
            DO K = 1,NOUT(ICNUM(N,IND1,NC))
              icanal_down = into(icnum(n,ind1,nc),k)
                if (icanal_down .eq. 0) then
                  itide_down= itide_down + 1
                  twdph_down= get_tide(ycnlnode(icnum(n,ind1,nc)
&                   ,itide_down),jday,proj_sea_level_rise)
                  twdph_down= 1.025*(twdph_down+2.0)-2.0
                  CRELEV_down = AMAX1(CREL(ICNUM(N,IND1,NC),K)
&                                 ,twdph_down)
                else
                  CRELEV_down = CREL(ICNUM(N,IND1,NC),K)
                endif
             IF(CNLSTG(N,IND1,NC)-stagediff
&                 .GT.CRELEV_down) THEN
             DSCHGDN =STRCF(ICNUM(N,IND1,NC),K)*GWDTH(ICNUM(N,IND1
&              ,NC),K)*(CNLSTG(N,IND1,NC)-stagediff-CRELEV_down)
&              * SQRT(CNLSTG(N,IND1,NC)-stagediff-CRELEV_down)
             TDSCHGDN = TDSCHGDN + DSCHGDN*86400.
             ENDIF
            ENDDO
           ENDIF
            TOT_PUMP = 0.0
            DO K = 1,NSTRCTR(ICNUM(N,IND1,NC))
              if (nodcr(icnum(n,ind1,nc)).gt.1) then
                stagedff = float(nodcr(icnum(n,ind1,nc))-nodepos_su
&               (icnum(n,ind1,nc),k))/float(nodcr(icnum(n,ind1,nc))
&               -1) * hdc(icnum(n,ind1,nc),iseas)
              else
                stagedff = 0.0
              endif
              DSCHGDN = STROUT2(CNLSTG(N,IND1,NC)-stagedff
&                 ,ICNUM(N,IND1,NC),ICNUM(N,IND1,NC),K,0
&                 ,CRMIN,VOLWTR,PREVQU,RAIN,ITER,TOT_PUMP,JDAY
&                 ,tempf_volwtr,tempf_weircff,0)
              TDSCHGDN = TDSCHGDN + DSCHGDN*86400.
            ENDDO
             IF (ICNUM(N,IND1,NC) .LT. N) THEN
                QINFLWDN = PREVQU(ICNUM(N,IND1,NC))
             ELSE
                QINFLWDN = QU(ICNUM(N,IND1,NC))
             ENDIF
             AVVSPOVDN= -CGSEEP(ICNUM(N,IND1,NC),CNLSTG(N,IND1,NC)
&            -stagediff,0,0.)-TOVFLC(ICNUM(N,IND1,NC),CNLSTG(N,IND1
&            ,NC)-stagediff,0,0.,ITER,2) - QINFLWDN + TDSCHGDN +
&            (CNLSTG(N,IND1,NC)-stagediff-CHDEP(ICNUM(N,IND1,NC)))*
&            CL(ICNUM(N,IND1,NC))*WIDTH(ICNUM(N,IND1,NC))
             AVVOLCDN = AMAX1(AVVSPOVDN,0.0)/86400.
```

```
                 TVOLMXDN = TVOLMXDN + AVVOLCDN
               ENDDO
                CAPAC = AMIN1(CAPAC,TVOLMXDN)
             ENDIF
C
         IF ((IPURPSE(N,IND1,NS) .EQ. 2 .OR. IPURPSE(N,IND1,NS) .EQ. 3)
     &   .AND. IDNCLIDX .NE. M) THEN
            CAPACFLF(IMGT) = CAPAC
         ELSE
            CAPACFLF(IMGT) = 0.0
         ENDIF
         IF (IPURPSE(N,IND1,NS) .EQ. 1 .OR. IPURPSE(N,IND1,NS) .EQ. 3)
     &   THEN
            CAPACWSF(IMGT) = CAPAC + CAPACWSF(IMGT)
         ENDIF
C
         IF ( ISPEC(N,IND1) .EQ. 0) THEN
            IF( LOGDISGF) THEN
C
               IF (IRATING(N,IND1,NS,IMGT) .EQ. 1) THEN
                  IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
                    COUTFLF = CAPACFLF(IMGT)
                  ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                    PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,
     &              IND1,ISEAS))**3*(STAGEM - RMNSTGS(N,IND1,ISEAS))
     &              **3
                    COUTFLF = PCTGO * CAPACFLF(IMGT)
                  ELSE
                    COUTFLF = 0.0
                  ENDIF
               ELSE
                  COUTFLF = CAPACFLF(IMGT)
               ENDIF
C
            ELSE
               COUTFLF = 0.0
            ENDIF
            IF (flnm(iflo).eq.'S335 ' .and. nint(s337fc).gt.0) COUTFLF
     &        = min(COUTFLF,s337fc)
            IF (.NOT. LFLWTRG(ids_trg_idx(N,IND1))) THEN
               call match (flnm(iflo),1,6,struc_name_w_targ ,
     &         no_of_struc_w_ds_targ_no_res,6,istruc_targ_indx)
               COUTFLF = COUTFLF * frc_flow_targ(istruc_targ_indx)
            ENDIF
C
            STROUT = COUTFLF + STROUT
            IF (FLNM(IFLO) .EQ. strname_spec(22))COUTFL332 = COUTFLF
            IF (FLNM(IFLO) .EQ. strname_spec(23))COUTFL18C = COUTFLF
         ENDIF
   20 CONTINUE
      IF (ISPEC(N,IND1) .EQ. 0) THEN
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF ( FLNM(IFLO) .EQ. strname_spec(22) .OR. FLNM(IFLO) .EQ.
     &   strname_spec(23)) THEN
C
            IF (FLNM(IFLO) .EQ. strname_spec(22)) THEN
```

```
               IF (opt_flow_to_ts .EQ. 'TSMINDL') THEN
                  WSDMND = VOL332(MONTH) - COUTFL332
               ELSE
                  WSDMND = VOL332DO(MONTH) - COUTFL332
               ENDIF
               WSDMND = AMAX1(WSDMND,0.0)
               CAPCWS = amax1(DISCOEF(N,IND1,1,1)-COUTFL332,0.0)
               COUTWS = AMIN1(WSDMND,CAPCWS)
            ELSE
               WSDMND = AMAX1(VOL18C(MONTH)/86400.,DQU(IDNCLIDX)) -
     &         COUTFL18C
               WSDMND = AMAX1(WSDMND,0.0)
               CAPCWS = amax1(DISCOEF(N,IND1,1,1)-COUTFL18C,0.0)
               COUTWS = AMIN1(WSDMND,CAPCWS)
            ENDIF
         ELSE
            WSDMND = amax1(tdqu_dn(IDNCLIDX) - COUTFLF,0.0)
            rem_capacws = amax1(CAPACWSF(IMGT) - COUTFLF,0.0)
            COUTWS = AMIN1(rem_capacws,WSDMND)
C           if (iactual_flow_sim .eq. 1) print *,COUTWS,COUTFLF
C    $      ,rem_capacws,CAPACWSF(IMGT),upstream_inflow_lim,' lll'
         ENDIF
C
         STROUT = STROUT + COUTWS
C
C  CALCULATE MAXIMUM AVAILABLE STORAGE IN RESERVOIR RECEIVING DISCHARGE
C   FROM STRUCTURE,IF APPLICABLE
C
         DIFFDRS = 0.0
         COUTRS = STROUT
         TCOUTRS = 0.0
         IF (NRES(N,IND1) .GT. 0) THEN
            DO IR = 1,NRES(N,IND1)
               COUTTP = PCTRES(N,IND1,IR) * STROUT + DIFFDRS
               COUTRS = AMIN1(COUTTP,VOLWTR(IRSRCVR(N,IND1,IR)))
               DIFFDRS = STROUT * PCTRES(N,IND1,IR) - COUTRS
               TCOUTRS = TCOUTRS + COUTRS
            END DO
            DO IR = 1,NRES(N,IND1)
               IF (PCTRES(N,IND1,IR) .LT. 0.010) THEN
                  icres_dn = 0
                  DO j = 1,nstrctr(idnclidx)
                     IF (nres(idnclidx,j) .GT. 0) THEN
                        DO irr = 1,nres(idnclidx,j)
                           IF (irsrcvr(idnclidx,j,irr) .EQ. IRSRCVR
     &                     (N,IND1,IR)) THEN
                              icres_dn = icres_dn + 1
                           ENDIF
                        END DO
                     ENDIF
                  END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                  IF (icres_dn .NE. 0) THEN
                     IF (resname(IRSRCVR(N,IND1,IR)) .EQ.
     &               name_res_spec_code(15)) THEN
                        temp_availstor = tempf_volwtr(IRSRCVR(N,IND1,IR)
     &                  ) - outf_from_c11res
```

```
                        temp_availstor = amax1(temp_availstor,0.0)
                        temp_struccap = tempf_weircff(IRSRCVR
     &                  (N,IND1,IR) ,i_res_inf_struc_indx(n,ind1,ir)) -
     &                  outf_from_c11res
                     ELSE
                        temp_availstor = tempf_volwtr(IRSRCVR
     &                  (N,IND1,IR))
                        temp_struccap = tempf_weircff(IRSRCVR
     $                  (N,IND1,IR) ,i_res_inf_struc_indx(n,ind1,ir))
                     ENDIF
                     strout = amin1(strout,temp_availstor,temp_struccap)
                     tempf_volwtr(IRSRCVR(N,IND1,IR)) = tempf_volwtr
     $               (IRSRCVR(N,IND1,IR)) - STROUT/NRES(N,IND1)

                     tempf_weircff(IRSRCVR(N,IND1,IR),
     $               i_res_inf_struc_indx (n,ind1,ir)) = tempf_weircff
     $               (IRSRCVR(N,IND1,IR) ,i_res_inf_struc_indx
     $               (n,ind1,ir)) - STROUT/NRES(N,IND1)
                  ELSE
                     IF (resname(IRSRCVR(N,IND1,IR)) .EQ.
     $               name_res_spec_code(15)) THEN
                        temp_availstor = volwtr(IRSRCVR(N,IND1,IR)) -
     $                  outf_from_c11res
                        temp_availstor = amax1(temp_availstor,0.0)
                     ELSE
                        temp_availstor = volwtr(IRSRCVR(N,IND1,IR))
                     ENDIF
                     strout = amin1(strout,temp_availstor)
                  ENDIF
               ELSE
                  strout = TCOUTRS
               ENDIF
            END DO
         ENDIF
      ELSE
C
C-----------------------------------------------------------------
C          GAPS IN L-5 TO TRANSFER WATER FROM HOLEY LAND TO WCA3
C-----------------------------------------------------------------
C
         GWDPTH = (ELLS(NODEHL) - H(NODEHL)) * S(NODEHL)
         IF (POND(NODEHL) .GT. GWDPTH) THEN
            HLSTAGE = POND(NODEHL) - GWDPTH + ELLS(NODEHL)
         ELSE
            HLSTAGE = (POND(NODEHL) - GWDPTH)/S(NODEHL) + ELLS(NODEHL)
         ENDIF
         call bpts(iday,month,10.0,7,1,1,holy_sched)
         IF (FLNM(IFLO) .EQ. strname_spec(26) .OR. FLNM(IFLO) .EQ.
     $   strname_spec(27) .OR. FLNM(IFLO) .EQ. strname_spec(28)) THEN
            IF (hlyenv .EQ. 'TRUE ') THEN
               holy_stage = stage_import_wca(iholy)
            ELSE
               holy_stage = holy_sched
            ENDIF
            IF (.NOT. DRYSEAS .OR. HLSTAGE .GE. holy_stage + 1.0) THEN

C
```

```
               STROUT = CAPACFLF(2)
            ELSE
               STROUT = CAPACFLF(1)
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C                   PUMP FROM C51W TO PROP L8 BPIT RES
C-----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(30)) THEN
            STROUT = MIN(tot_avl_l8_bpit_res
     $      ,DISCOEF(N,IND1,1,1)-rl8runoff_to_bptres
     $      ,VOLWTR(irsrcvr(n,ind1,1)))

C-----------------------------------------------------------------
C                            L31N PUMP STATION
C-----------------------------------------------------------------
C
C
         ELSEIF (char_struc_name(1:5) .eq. 'S356 ') THEN
            IF (KFLO(IFLS334) .NE. 0 .OR. DQU(L31S) .GT. 0.0) THEN
               CAPACFLF(ISEAS) = 0.0
            ELSE
               CAPACFLF(ISEAS) = DISCOEF(N,IND1,1,1)
     &                  * frac_struc_cap(n,ind1,month)
            ENDIF
              DIFFS = RMXSTG(N,IND1,ISEAS)-RMNSTGS(N,IND1,ISEAS)
              IF (STALEM .LT. RMNSTGS(N,IND1,ISEAS)+DIFFS .AND.
     &         STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                QPUMP = (STAGEM - RMNSTGS(N,IND1,ISEAS)) *(1.0/DIFFS)
     &            * DISCOEF(N,IND1,1,1) * frac_struc_cap(n,ind1,month)
              ELSEIF (STAGEM .GE. RMNSTGS(N,IND1,ISEAS)+DIFFS) THEN
                QPUMP = DISCOEF(N,IND1,1,1)
     &            *frac_struc_cap(n,ind1,month)
              ELSE
                QPUMP = 0.0
              ENDIF
            AVVOL_UP = (STAGEM-RMNSTGS(N,IND1,ISEAS))*CL(N)*WIDTH(N) +
     $        CGSEEP(N,RMNSTGS(N,IND1,ISEAS),0,0.)+TOVFLC(N,
     $         RMNSTGS(N,IND1,ISEAS),0,0.,ITER,1) + QU(N)
            AVVOL_UP = AMAX1(AVVOL_UP,0.0)/86400.
            if (iseep_lim_s356_opt .ne. 0) then
               AVVOL_UP_S = AMAX1(QU(N) - KFLO(IFLS335)*86400.
     $           - KFLO(IFLS334)*86400.- s334_fc*86400.,0.0)
               AVVOL_UP = amin1(AVVOL_UP_S/86400.,AVVOL_UP)
            endif
            QPUMPT =CAPACFLF(ISEAS)
            STROUT =AMIN1(QPUMPT,DISCOEF(N,IND1,1,1),AVVOL_UP,QPUMP)
            if (irceive(n,ind1) .eq. 1) then
             if ( lp(17)) then
              if (chdep(IDNCLIDX) .gt. CNLSTG(N,IND1,1)-0.25) then
                frac_max_disc_c = (CNLSTG(N,IND1,1)-chdep(idnclidx))
     $           / 0.25
                frac_max_disc_c = amax1(frac_max_disc_c,0.0)
              else
                frac_max_disc_c = 1.0
              endif
             else
```

```
              frac_max_disc_c = 1.0
             endif
            else
              frac_max_disc_c = 1.0
              if (stage3b .gt. 8.5 .or. nint(rmnstg_dual_ops0(n,ind1))
     $            .eq. 99) then
                strout = 0.0
              endif
            endif
            DPTHGW = (ELLS(NODE_G3273) - H(NODE_G3273)) * S(NODE_G3273)
            IF (POND(NODE_G3273) .GT. DPTHGW) THEN
              STAGE_G3273 = POND(NODE_G3273) - DPTHGW + ELLS(NODE_G3273)
            ELSE
              STAGE_G3273 = (POND(NODE_G3273)-DPTHGW)/S(NODE_G3273)
     $                          + ELLS(NODE_G3273)
            ENDIF
            IF (opt_mrp_to_enp.and.irceive(n,ind1).eq.1) THEN
             DO ncal = 1,nch
              DO i=1,nstrctr(ncal)
                IF (tmpname(ncal,i).eq.'S333  ') THEN
                   istruc_s333=i
                   incl_s333 = ncal
                ENDIF
              END DO
             ENDDO
              if (STAGE_G3273 .gt. stg_trigger_fl(incl_s333,istruc_s333,
     $          1)) then
                frac_max_disc_gc = 0.0
              else
                frac_max_disc_gc = 1.0
              endif
            else
              frac_max_disc_gc = 1.0
            endif
            STROUT = STROUT * amin1(frac_max_disc_gc,frac_max_disc_c)
C-----------------------------------------------------
C                       S356A
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(32)) THEN
            IF (KFLO(IFLS334) .NE. 0 .OR. nint(DQU(L31S)) .GT. 0 .OR.
     $      nint(DQU(IC4)) .GT. 0) THEN
               CAPACFLF(ISEAS) = 0.0
            ELSE
               CAPACFLF(ISEAS) = DISCOEF(N,IND1,1,1)
            ENDIF
            QPUMPT =CAPACFLF(ISEAS)
            STROUT   =AMIN1(QPUMPT,DISCOEF(N,IND1,1,1))
C
C-------------------------------------------------------------
C             WATER SUPPLY FROM L-8 TO M-CANAL
C-------------------------------------------------------------
C
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(33)) THEN
            HDIFF = ABS(STAGEM - CHDEP(IDNCLIDX))
            IF (STAGEM .LT. CHDEP(IDNCLIDX)) THEN
               IF (STAGEM .GT. 10.2) THEN
                  CAPAC = 7. 15* (6.0 - HDIFF) + 182.0
```

```
               ELSEIF (STAGEM .GE. 9.5) THEN
                  IF (HDIFF .GT. 11.5) THEN
                     CAPAC = 136.0 - 10.29* (HDIFF - 11.5)
                  ELSEIF (HDIFF .GT. 4.0) THEN
                     CAPAC = 136.0 + 4.53 * (11.5 - HDIFF)
                  ELSE
                     CAPAC = 170.0
                  ENDIF
               ELSEIF (STAGEM .GT. 9.1) THEN
                  CAPAC = 150.0* (STAGEM -9.1) / 0.4
               ELSE
                  CAPAC = 0.0
               ENDIF
            ELSE
               IF (STAGEM .GE. 9.5) THEN
                  CAPAC = 170.0
               ELSEIF (STAGEM .GT. 9.1) THEN
                  CAPAC = 170.0 * (STAGEM - 9.1) / 0.4
               ELSE
                  CAPAC = 0.0
               ENDIF
            ENDIF
            CAPAC = CAPAC*FACTS2MC
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            GWDPTH = (ELLS(IWPBCAT)-H(IWPBCAT))*S(IWPBCAT)
            IF (POND(IWPBCAT) .GT. GWDPTH) THEN
               STAGE = POND(IWPBCAT) - GWDPTH + ELLS(IWPBCAT)
            ELSE
               STAGE = (POND(IWPBCAT) - GWDPTH)/S(IWPBCAT) + ELLS
     $         (IWPBCAT)
            ENDIF
C
            IF (STAGEM .GT. rmin_l8stg_flow_to_mcnl(iseas)) THEN
               IF (STAGE .LT. rmax_wpbcat_for_inflw) THEN
                  stg_rmx = rmax_wpbcat_for_inflw
                  DISHGDN = AMIN1(DQU(ILKMG),50.0)
                  AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $            (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $            (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN
     $            *86400.
                  AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
                  capac_out = capac
C        IF (stagem .GT. rmin_l8stg_flow_to_mcnl(iseas) + 0.25) THEN
C          capac_out = capac
C        ELSE
C          capac_out = 4.0*capac*(stagem-rmin_l8stg_flow_to_mcnl(iseas))
C        ENDIF
                  STROUT = AMIN1(CAPAC_OUT,AVVOLDN)
                  IF (STAGELO.LT.SCHED(IDAY,MONTH)) STROUT=AMIN1(CAPAC ,
     $            DQU(ILKMG)+DMNDAGMC,50.0)
               ELSE
                  STROUT = 0.0
               ENDIF
            ELSE
               stg_rmx = CRMIN(IDNCLIDX,iseas)
               DISHGDN = AMIN1(DQU(ILKMG),50.0)
```

```
               AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $         (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $         (IDNCLIDX,stg_rmx, 0,0.,iter,2) - QINFLOWN(N) + DISHGDN
     $         *86400.
               AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
               STROUT = AMIN1(CAPAC,AVVOLDN)
               IF (STAGELO.LT.SCHED(IDAY,MONTH)) STROUT=AMIN1(CAPAC ,DQU
     $         (ILKMG)+DMNDAGMC,50.0)
            ENDIF
C-----------------------------------------------------
C        PUMP TO C9RES FROM US27S
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(83)) THEN
C

            rem_capac = amax1(capacflf(imgt) - outf_to_c9res_via_us27s,
     $      0.0)
            rem_pump_cap = DISCOEF(n,ind1,1,1) - outf_to_c9res_via_us27s
            rem_avvol = VOLWTR(IRSRCVR(N,IND1,1))
            strout = amin1(rem_avvol,rem_capac,rem_pump_cap)
C-----------------------------------------------------
C            S-9
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(85)) THEN
C
            strout = capacflf(imgt)
C-----------------------------------------------------
C            S-9A
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(86)) THEN
          IF (nint(outf_s9_l) .eq. 0) THEN
            strout=capacflf(imgt)
          ELSE
            strout=0.0
          ENDIF
c         print *,outf_s9_l,strout,capacflf(imgt),'  S9ALIM'
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-------------------------------------------------------------
C          PROPSED STRUC IN L-37 Canal to PASS ENVIRONMENTAL FLOWS
C-------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(59)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))** 3* (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               strout = amax1(coutfl,prevqu(n)/86400.)
               strout = amin1(cout,CAPACFLF(IMGT))
            ELSE
               strout = coutfl
            ENDIF
```

```
C-----------------------------------------------------------
C          PROPSED STRUC IN L-30 to PASS ENVIRONMENTAL FLOWS
C-----------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(61)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))** 3* (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               strout = amax1(coutfl,prevqu(n)/86400.)
               strout = amin1(cout,CAPACFLF(IMGT))
            ELSE
               strout = coutfl
            ENDIF
C-------------------------------------------------
C              S-335
C-------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(6)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))**3 * (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               struc_inflw = amax1(prevqu(n)/86400.-tseepge(ilv30),0.0)
               strout = amax1(coutfl,struc_inflw)
               strout = amin1(strout,CAPACFLF(IMGT))
            ELSE
               strout = coutfl
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-------------------------------------------
C        M1 LOWER BASIN TO MCNL
C-------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(34)) THEN
C
C  downstream avail storage
            DISHGDN = 0.0
            TOT_PUMP = 0.0
            stg_rmx = rmax_wpbcat_for_inflw
            DISHGDN = AMIN1(DQU(ILKMG),50.0)
            AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $      (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $      (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN*86400.
            AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
C
C  calculate outflow
            STROUT = amin1(avvoldn,DISCOEF(N,IND1,1,1))
```

```
C-----------------------------------------
C        M1 LOWER BASIN TO MCNL
C-----------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(35)) THEN
C
C  downstream avail storage
            DISHGDN = 0.0
            TOT_PUMP = 0.0
            stg_rmx = rmax_wpbcat_for_inflw
            DISHGDN = AMIN1(DQU(ILKMG),50.0)
            AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $      (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $      (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN*86400.
            AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
C
C  calculate outflow
            STROUT = amin1(avvoldn,DISCOEF(N,IND1,1,1))
C
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(70)) THEN
            IF (stagem .LT. crel(n,1)) THEN
               IF (stagem .GT. RMNSTGS(N,IND1,ISEAS)+0.5) THEN
                  strout = capacflf(imgt)
                  IF (stagem .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                     strout = 2.0* capacflf(imgt)*(stagem - RMNSTGS(N,
     $               IND1,ISEAS))
                  ELSE
                     strout = 0.0
                  ENDIF
               ELSE
                  strout = 0.0
               ENDIF
            ELSE
               strout = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C              S142 BACK TO WCA3A
C-----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(39)) THEN
            IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
               S142CAP = DISCOEF(N,IND1,1,1)*(STAGEM-CHDEP(IDNCLIDX))
     $         **PWR(N,IND1,1,1)
            ELSE
               S142CAP = 0.0
            ENDIF
            STROUT = AMIN1(S142CAP,TFLG123)
C
C-----------------------------------------------------
C   PROP PUMP DIVERTING C11W OUTFLOW TO SOUTH
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(42)) THEN
            CAPAC = DISCOEF(N,IND1,1,1)
            strout = capac
C-----------------------------------------------------
C                 S-174
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(64)) THEN
```

```
            IF (IRATING(N,IND1,1,IMGT) .NE. 2) THEN
               IF (DELTAH .GT. RMINHD(N,IND1)) THEN
                  S174CAP = DISCOEF(N,IND1,1,IMGT)*(DELTAH - RMINHD
     $            (N,IND1)) ** PWR(N,IND1,1,IMGT)
               ELSE
                  S174CAP = 0.0
               ENDIF
            ELSE
               S174CAP = DISCOEF(N,IND1,1,IMGT)
            ENDIF
            s174cap = amin1(s174cap,STRDCAP(N,IND1,1,IMGT))
            IF (opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               IF (dqu(idnclidx) .GT. 0.0 .OR. LWSLO(itylsl)) THEN
                  IF (LWSLO(itylsl)) THEN
                     STROUT = s174cap
                  ELSE
                     STROUT = amin1(s174cap,dqu(idnclidx))
                  ENDIF
               ELSE
                  STROUT = 0.0
               ENDIF
            ELSE
               STROUT = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-------------------------------------------------------
C                 S-175
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(65)) THEN
            deltah = stagem - (chdep(idnclidx)+hdc(idnclidx,iseas))
            IF (DELTAH .GT. RMINHD(N,IND1)) THEN
               S175CAP = DISCOEF(N,IND1,1,IMGT) * (DELTAH - RMINHD
     $         (N,IND1)) ** PWR(N,IND1,1,IMGT)
            ELSE
               S175CAP = 0.0
            ENDIF
            IF ( opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               IF (main_pres_level_fl_prot_ts .EQ. 'FALSE') THEN
                  IF (.NOT. LWSLO(itylsl)) THEN
                     stg_min = stage_import_wca(il31w_min_rf) + 0.1
                     stg_max = stage_import_wca(il31w_min_rf) + 0.3
                  ELSE
                     stg_min = stage_import_wca(il31w_min_rf) + 0.3
                     stg_max = stage_import_wca(il31w_min_rf) + 0.7
                  ENDIF
               ELSE
                  stg_max = RMXSTG(N,IND1,ISEAS)
                  stg_min = RMNSTGS(N,IND1,ISEAS)
               ENDIF
               IF (STAGEM .GT. stg_max) THEN
                  STROUT = s175cap
               ELSE IF (STAGEM .GT. stg_min) THEN
                  STROUT = 1.0/(stg_max-stg_min)*s175cap*(STAGEM
     $            -stg_min)
               ELSE
                  STROUT = 0.0
               ENDIF
```

```
            ELSE
               STROUT = 0.0
            ENDIF
C-----------------------------------------------------
C               S-332
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(22)) THEN
            IF (s332sparrow_opt .EQ. 'TRUE ') THEN
               IF (month.EQ. 7 .AND. iday .GT. 15) THEN
                  frac_struc_cap_s332 = frac_struc_cap(n,ind1,month+1)
               ELSE
                  frac_struc_cap_s332 = frac_struc_cap(n,ind1,month)
               ENDIF
            ELSE
               frac_struc_cap_s332 = frac_struc_cap(n,ind1,month)
            ENDIF
            s332cap = DISCOEF(n,ind1,1,1)*frac_struc_cap_s332
            IF (opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               STROUT = s332cap
            ELSE
               STROUT = 0.0
            ENDIF
C-------------------------------------------------------
C                          S-197
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(68)) THEN
            IF (stagem .GT. rmnstgs(n,ind1,iseas)) THEN
               frac_go = (stagem-rmnstgs(n,ind1,iseas))** 2/ (rmxstg(n,
     $         ind1,iseas)-rmnstgs(n,ind1,iseas))**2
               frac_go = amin1(frac_go,1.0)
            ELSE
               frac_go = 0.0
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            tw197 = get_tide(ycnlnode(n,1),jday,proj_sea_level_rise)
            tw = 1.025*(tw197+2.0)-2.0

            IF (stagem .GT. tw) THEN
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
C          IF (conv .EQ. 1)
C    +     PRINT *,icnl_no_fc(n,ind1,ic),stg_trig_oper(n,ind1,ic,1)
                  IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $            (n,ind1,ic,1)) THEN
                     frac_full = 7.0/13.0 + ((min(chdep(icnl_no_fc
     $               (n,ind1,ic)) - stg_trig_oper(n,ind1,ic,1),0.1))**2
     $               /(0.1)**2)*(6.0/13.0)
                     cout = amin1(frac_full,frac_go) * DISCOEF
     $               (N,IND1,1,1) * (stagem - TW) ** PWR(N,IND1,1,1)
     $               * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,1),cout,chdep(n),'  AAA'
                     GO TO 30
                  ENDIF
               END DO
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
```

```
                  IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $            (n,ind1,ic,2)) THEN
                     frac_full = 3.0/13.0 + ((chdep(icnl_no_fc
     $               (n,ind1,ic)) - stg_trig_oper(n,ind1,ic,2))**2/(0.5
     $               *(stg_trig_oper(n ,ind1,ic,1)-stg_trig_oper(n,ind1,
     $               ic,2))**2))*(4.0/13.0)
                     frac_full = amin1(frac_full,7.0/13.0)
                     cout = amin1(frac_full,frac_go)*DISCOEF(N,IND1,1,1)
     $               * (stagem - TW) ** PWR(N,IND1,1,1)
     $               * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,2),cout,chdep(n),'  BBB'
                     GO TO 30
                  ENDIF
               END DO
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
                  IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $            (n,ind1,ic,3)) THEN
                     frac_full = ((chdep(icnl_no_fc(n,ind1,ic)) -
     $               stg_trig_oper(n,ind1,ic,3))**2/(0.5*(stg_trig_oper
     $               (n ,ind1,ic,2)-stg_trig_oper(n,ind1,ic,3))**2))
     $               *(3.0/13.0)
                     frac_full = amin1(frac_full,3.0/13.0)
                     cout = amin1(frac_full,frac_go)*DISCOEF(N,IND1,1,1)
     $               * (stagem - TW) ** PWR(N,IND1,1,1)
     $               * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,3),cout,chdep(n),'  CCC'
                     GO TO 30
                  ENDIF
               END DO
               cout_mx = frac_go * DISCOEF(N,IND1,1,1) * (stagem - TW)
     $         ** PWR(N,IND1,1,1) * frac_struc_cap(N,IND1,month)
               IF (nint(dqu(lc111)) .EQ. 0) THEN
                  cout = amin1(cout_mx,kflo(ifls176))
               ELSE
                  cout = 0.0
               ENDIF
            ELSE
               cout = 0.0
            ENDIF
   30       strout = cout
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------
C                 S-331 AND S173
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(67)) THEN
            DPTHGW = (ELLS(NODANGL) - H(NODANGL)) * S(NODANGL)
            IF (POND(NODANGL) .GT. DPTHGW) THEN
               STG_ANGELS1 = POND(NODANGL) - DPTHGW + ELLS(NODANGL)
            ELSE
               STG_ANGELS1 = (POND(NODANGL)-DPTHGW)/S(NODANGL) + ELLS
     $         (NODANGL)
            ENDIF
            DPTHGW = (ELLS(IG2517) - H(IG2517)) * S(IG2517)
```

```
            IF (POND(IG2517) .GT. DPTHGW) THEN
               STG_ANGELS2 = POND(IG2517) - DPTHGW + ELLS(IG2517)
            ELSE
               STG_ANGELS2 = (POND(IG2517)-DPTHGW)/S(IG2517) + ELLS
     $         (IG2517)
            ENDIF
            stg_angels = (stg_angels1 + stg_angels2)/2.0
            LANGELX = stg_angels .GE. stg_trigger_fl(n,ind1,3)
            IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
               str_cap(1) = DISCOEF(N,IND1,1,1) * (STAGEM-CHDEP
     $         (IDNCLIDX)) * PWR(N,IND1,1,1)
            ELSE
               str_cap(1) = 0.0
            ENDIF
C
            LANGEL_MXX = .NOT.(stg_angels .LT. stg_trigger_fl(n,ind1,2))
            IF (stg_angels .GT. stg_trigger_fl(n,ind1,1).OR. LANGEL_MXX)
     $      THEN
               LANGEL_MXX = .TRUE.
               stg_min = stg_min_fc(n,ind1,2)
            ELSE IF (stg_angels .GT. stg_trigger_fl(n,ind1,3)) THEN
               stg_min = stg_min_fc(n,ind1,1)
            ELSE
               IF (runmode .EQ. 'CALIB') THEN
                  stg_min = 5.0
               ELSE
                  stg_min = 5.2
               ENDIF
            ENDIF
            do ll = 1,idrwn
              if (n .eq. icanal_no_drawdown(ll)) then
                idrawdwnc = 1
              endif
            enddo
            if (nstorm_day(ndays_sim_sd).eq.1 .and. idrawdwnc.eq.1)
     $       then
               stg_min = stg_min - 1.0
            endif
            idualopsc = 0
            do ll = 1,ndual_canal_ops(1)
              if (n .eq. icanal_no_dualops(ll,1)) then
                 idualopsc = 1
              endif
            enddo
            if (nstorm_day(ndays_sim_sd).ne.1 .and. idualopsc.eq.1) then
               stg_min = stg_min - 0.5
            endif
C
            COUT_FC = 0.0
            DO ns = 1,nsubstr(n,ind1)-1
               IF (ipurpse(n,ind1,ns) .NE. 1) THEN
                  IF( LOGDISGF) THEN
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                     IF (LANGELX) THEN
                        IF (STAGEM .GT. stg_min + 0.4) THEN
                           COUTFLF = str_cap(ns)
```

```
                        ELSEIF(STAGEM .GT. stg_min) THEN
                           PCTGO = 6.25* (STAGEM - stg_min)**2
                           COUTFLF = PCTGO * str_cap(ns)
                        ELSE
                           COUTFLF = 0.0
                        ENDIF
C              ELSE
                        IF (STAGEM .GT. 6.2) THEN
                           COUTFLF = str_cap(ns)
                        ELSEIF(STAGEM .GT. 6.0) THEN
                           PCTGO = 25.0* (STAGEM - 6.0)**2
                           COUTFLF = PCTGO * str_cap(ns)
                        ELSE
                           COUTFLF = 0.0
                        ENDIF
                     ENDIF
                  ELSE
                     COUTFLF = 0.0
                  ENDIF
               ELSE
                  COUTFLF = 0.0
               ENDIF
               COUT_FC = COUT_FC + COUTFLF
            END DO
            ilast_subs = nsubstr(n,ind1)
            IF (INGVHDOPT(N,IND1,ilast_subs) .EQ. 1) THEN
               IF (STAGEM .LE. TW) THEN
                  CAPACP = DISCOEF(N,IND1,ilast_subs,IMGT)
     $             * frac_struc_cap(N,IND1,month)
               ELSE
                  CAPACP = 0.0
               ENDIF
            ELSE
               CAPACP = DISCOEF(N,IND1,ilast_subs,IMGT)
     $             * frac_struc_cap(N,IND1,month)
            ENDIF
            IF (ipurpse(n,ind1,ilast_subs) .NE. 1) THEN
               IF (LOGDISGF) THEN
                  CAPAC = CAPACP
               ELSE
                  CAPAC = 0.0
               ENDIF
            ELSE
               CAPAC = 0.0
            ENDIF
            str_cap(ilast_subs) = CAPAC
            STROUT = cout_fc + str_cap(ilast_subs)
C-----------------------------------------------------------------
C  PROPOSED PUMP : C-18 TO LOX SLOUGH
C-----------------------------------------------
         ELSE IF (flnm(iflo) .EQ. 'LXSLWS') THEN
            IF (loxenv .EQ. 'FALSE') THEN
               strout = 0.0
            ELSE
               IF (LWSLO(iloxah)) THEN
                  strout = DISCOEF(N,IND1,1,1)
               ELSE
```

```
                  strout = 0.0
               ENDIF
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C             WATER SUPPLY FROM CATCHMENT AREA TO  EASTERN
C               REACH OF M - CANAL
C----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(38)) THEN
            IF (STAGEM .GT. 14.5) THEN
               RMXDSCHG = DISCOEF(N,IND1,1,1)
            ELSEIF(STAGEM .GT. 12.5) THEN
               RMXDSCHG = 12.5* (STAGEM-12.5) + 25.0
            ELSE
               RMXDSCHG = 25.0
            ENDIF
            STROUT = AMIN1(DQU(ILKMG),RMXDSCHG)
C-------------------------------------------------
C             BACKFLOW FROM L-8 INTO LAKE OKEECHOBEE
C-------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(37)) THEN
C
            IF (STAGEM+HDC(N,iseas) .GT. STAGELO) THEN
C
               STROUT = CAPACFLF(1)
            ELSE
               STROUT = 0.0
            ENDIF
C----------------------------------------------------------------
C                STRUCTURE TO ALLOW BACK FLOW BY
C                 GRAVITY FROM M-CANAL TO L-8
C----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(36)) THEN
            rmin_stg_fc = rmnstgs(n,ind1,iseas)
            IF (STAGEM+HDC(N,iseas) .GT. rmin_stg_fc + 1.0) THEN
               STROUT = DISCOEF(N,IND1,1,1)
            ELSEIF (STAGEM+HDC(N,iseas) .GT. rmin_stg_fc) THEN
               STROUT = 1. 0* DISCOEF(N,IND1,1,1) * (STAGEM + HDC
     $         (N,iseas) - rmin_stg_fc)
            ELSE
               STROUT = 0.0
            ENDIF
C
C-----------------------------------------------------------------
C             WATER SUPPLY FROM EASTERN M-CANAL TO CLEAR LAKE
C----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(54)) THEN
            IF (STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               COUTWS = DQU(ILKMG)
               STROUT = COUTWS
            ELSE
               IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
                  CAPACITY = DISCOEF(N,IND1,1,1)*(STAGEM-CHDEP(IDNCLIDX)
     $            ) ** PWR(N,IND1,1,1)
               ELSE
                  CAPACITY = 0.0
```

```
               ENDIF
               STROUT = AMIN1(DQU(ILKMG),CAPACITY)
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------
C                          S-316
C                       PROPOSED STRCTR
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(69)) THEN
            IF (DQU(IDNCLIDX) .GT. 0.0) THEN
               STGDN = CHDEP(IDNCLIDX)+0.1
               AVVOL_UP = (STAGEM-STGDN) *CL(N)*WIDTH(N) + CGSEEP(N,
     $         STGDN,0,0.)+TOVFLC(N,STGDN,0,0.,iter,2)
               AVVOL_UP = AMAX1(AVVOL_UP/86400.,0.0)
               STROUT = AMIN1(AVVOL_UP,DISCOEF(N,IND1,1,1),DQU(IDNCLIDX)
     $         )
            ELSE
               AVVOL_UP = (STAGEM-RMNSTGS(N,IND1,ISEAS))*CL(N)*WIDTH (N)
     $         +CGSEEP(N,RMNSTGS(N,IND1,ISEAS),0,0.)+TOVFLC(N, RMNSTGS
     $         (N,IND1,ISEAS),0,0.,iter,2)
               AVVOL_UP = AMAX1(AVVOL_UP/86400.,0.0)
               STROUT = AMIN1(AVVOL_UP,DISCOEF(N,IND1,1,1))
            ENDIF
C
C-------------------------------------------------------------
C                 S-319 PROPOSED PUMP FOR C51W
C-------------------------------------------------------------
         ELSEIF(FLNM(IFLO) .EQ. strname_spec(55)) THEN
            out_flow_east = 0.0
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
               capac_s155a = 625.0* (stagem-(chdep(ic51) +hdc
     $         (ic51,iseas))) ** 0.5
            ELSE
               capac_s155a = 0.0
            ENDIF
            cout = 0.0
            TDQU = AMAX1(DQU(IC51) - out_flow_east,0.0)
            WSDEL = TDQU
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
               COUT = COUT + amin1(WSDEL,capac_s155a)
            ENDIF
            out_flow_east = out_flow_east + amin1(WSDEL,capac_s155a)
            capac_s155a = capac_s155a - amin1(WSDEL,capac_s155a)
            COUT = WEIRCFF(5,1,1) + COUT
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
               COUT = COUT + capac_s155a
               out_flow_east = out_flow_east + capac_s155a
               capac_s155a = 0.0
            ENDIF
            STROUT = COUT
         ELSE
            STROUT = 0.0
         ENDIF
      ENDIF

      RETURN
```

```
      END
C*****************************************************************************
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      REAL FUNCTION strout2 (stagem, m, n, ind1, inflowopt, crmin,
     & volwtr, prevqu, rain, iter, tot_pump, jday, tempf_volwtr,
     & tempf_weircff, iactual_flow_sim)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'cnlneeds.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'dualops.inc'

      DIMENSION capacflf(max_n_seas), capacwsf(max_n_seas),
     & crmin(max_ncnls,max_n_seas),
     & prevqu(max_ncnls), qinflown(max_ncnls), rain(max_ncells),
     & str_cap(5), tempf_volwtr(max_n_res),
     & tempf_weircff(max_n_res,max_nostr_res), volwtr(max_n_res)

      LOGICAL langel_mxx, langelx, ldishgf, logdisgf

      CHARACTER char_struc_name*6

      DOUBLE PRECISION stagem,crmin,stg_rmx,STGDN,rmin_stg_fc,tw

      IFLO = ISTR(N,IND1)
      char_struc_name = flnm(iflo)
      IDNCLIDX = IDNCNL(N,IND1)
      IF (ITER .EQ. 0) THEN
         QINFLOWN(N) = PREVQU(N)
      ENDIF
C
      IDSHGF = 0
      IF (NDSCLIM(N,IND1) .GT. 1) THEN
         DO 10 I = 1,NDSCLIM(N,IND1)
            IF (CHDEP(ICNUM(N,IND1,I)) .GE. CNLSTG(N,IND1,I)) THEN
               IDSHGF = IDSHGF + 1
            ENDIF
   10    CONTINUE
      ENDIF
      LDISHGF = IDSHGF .EQ. NDSCLIM(N,IND1)
C
```

```
      IF (NDSCLIM(N,IND1) .GT. 1) THEN
         IF (FLNM(IFLO) .EQ. strname_spec(13)) THEN
            LOGDISGF = LDISHGF
         ELSE
            LOGDISGF = .NOT. LDISHGF
         ENDIF
      ELSE
         LOGDISGF = .TRUE.
      ENDIF
C
C
      IF (IRCEIVE(N,IND1) .EQ. 1) THEN
         IF (IDNCLIDX .LE. NCH) THEN
            STGDIFF_CANAL_TW = FLOAT(NODCR(IDNCLIDX)-NODEPOS_SD(N ,IND1)
     $      ) /FLOAT(NODCR(IDNCLIDX)) * HDC(IDNCLIDX,iseas)
            TW= CHDEP(IDNCLIDX) + STGDIFF_CANAL_TW
         ENDIF
         IF (FLNM(IFLO) .EQ. strname_spec(37).OR.FLNM(IFLO) .EQ.
     $   strname_spec(21)) TW = STAGELO
         IF (icell_no_sd(n,ind1) .gt. 0) THEN
           IF (lutyp(icell_no_sd(n,ind1)) .eq. 0) then
              twdph=get_tide(ycnlnode(n,1),jday,
     &         proj_sea_level_rise)
              tw = 1.025*(twdph + 2.0) - 2.0
           ENDIF
         ENDIF

      ELSEIF (IRCEIVE(N,IND1) .EQ. 2 .OR. IRCEIVE(N,IND1) .EQ. 3) THEN

         TW = ELLS(NODRCV(N,IND1,1)) + POND(NODRCV(N,IND1,1))
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
      IF (NCOEFF(N,IND1) .EQ. 1) THEN
         IMGT = 1
      ELSE
         IMGT = ISEAS
      ENDIF
      STROUT2 = 0.0
C
      DO JJ = 1,NCOEFF(N,IND1)
         CAPACFLF(JJ) = 0.0
         CAPACWSF(JJ) = 0.0
      END DO
C
      DO 20 NS = 1,NSUBSTR(N,IND1)
C
         IF (IRATING(N,IND1,NS,IMGT) .EQ. 1) THEN
            IF (FLNM(IFLO) .EQ. strname_spec(37)) THEN
              DELTAH = STAGEM + HDC(N,iseas) - TW
            ELSE
              DELTAH = STAGEM - TW
            ENDIF
            IF (DELTAH .GT. RMINHD(N,IND1)) THEN
               CAPAC =DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n,ind1,
     $         month) *(DELTAH - RMINHD(N,IND1))** PWR(N,IND1,NS,IMGT)
```

```
            ELSE
               CAPAC = 0.0
            ENDIF
            IF (ides_cap_limit(n,ind1) .EQ. 1) THEN
               CAPAC = AMIN1(CAPAC,STRDCAP(N,IND1,NS,IMGT))
            ENDIF
            IF (stagem.gt.rmxstg(n,ind1,iseas).or.lgrav(n,ind1,ns)) THEN
               IF (.not. lgrav(n,ind1,ns)) THEN
                  IF (stagem.gt.rmxstg(n,ind1,iseas)+0.2) THEN
                     capacfldc0=capac
                  ELSE
                     capacfldc0=5.0*capac*(stagem-
     &               rmxstg(n,ind1,iseas))
                  END IF
               ELSE
                 capacfldc0 = capac
               ENDIF
            ELSE
              capacfldc0 = 0.0
            ENDIF
            capac = capacfldc0
c           if (flnm(iflo).eq.'G211 ') print *,stagem,
c    +      DISCOEF(N,IND1,NS,IMGT),tw,CAPAC,'  G211LIM  '
c    +      ,month,iday
C
         ELSEIF (IRATING(N,IND1,NS,IMGT) .EQ. 2) THEN
            IF (iactual_flow_sim .eq. 0) THEN
               IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS).OR.LPUMP(N,IND1
     $         ,NS)) THEN
                  IF (.NOT. LPUMP(N,IND1,NS)) THEN
                     IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS) + 0.2) THEN
                        CAPAC = DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n
     $                  ,ind1,month)
                     ELSE
                        CAPAC = 5.0 * DISCOEF(N,IND1,NS,IMGT) *
     $                  frac_struc_cap(n,ind1,month) * (STAGEM -
     $                  RMXSTG(N,IND1,ISEAS))
                     ENDIF
                  ELSE
                     RMINSTG = RMNSTGS(N,IND1,ISEAS)
                     IF (STAGEM .GT. RMINSTG + 0.2) THEN
                        CAPAC=DISCOEF(N,IND1,NS,IMGT)
     $                  * frac_struc_cap(n,ind1,month)
                     ELSEIF(STAGEM .GT. RMINSTG) THEN
                        CAPAC=5.0*DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap
     $                  (n,ind1,month)*(STAGEM-RMINSTG)
                     ELSE
                        CAPAC=0.0
                     ENDIF
                  ENDIF
                  IF (LPUMP(N,IND1,NS)) THEN
                     AVVOLQ = (STAGEM - RMNSTGS(N,IND1,ISEAS)) * CL(N)
     $                 * WIDTH(N) + CGSEEP(N,RMNSTGS(N,IND1,ISEAS),0,0.)
     $                 + TOVFLC(N,RMNSTGS(N,IND1,ISEAS),0,0.,iter,1)
     $                 + AMAX1(QINFLOWN(N),0.0)
                     AVVOLQ     = AMAX1(AVVOLQ,0.0)/86400.
                     CAPAC     = AMIN1(AVVOLQ,CAPAC)
```

```
                 ELSE
                    AVVOLQ=(STAGEM-RMXSTG(N,IND1,ISEAS))*CL(N)*WIDTH(N)
     $              +CGSEEP(N,RMXSTG(N,IND1,ISEAS),0,0.)+TOVFLC(N
     $              ,RMXSTG(N,IND1,ISEAS),0,0.,iter,1)+AMAX1(QINFLOWN
     $              (N),0.0)
                    AVVOLQ    = AMAX1(AVVOLQ,0.0)/86400.
                    CAPAC     = AMIN1(AVVOLQ,CAPAC)
                 ENDIF
                 TOT_PUMP = TOT_PUMP + CAPAC
              ELSE
                AVVOLQ = 0.0
                CAPAC  = 0.0
              ENDIF
           ENDIF
        ELSEIF (IRATING(N,IND1,NS,IMGT) .EQ. 3) THEN
           IF (DQU(IDNCLIDX) .GT. 0.0) THEN
              TW = CRMIN(IDNCLIDX,iseas)
           ELSE
              TW = RMNSTGS(N,IND1,ISEAS)
           ENDIF
C
           IF (STAGEM .GT. TW) THEN
              CAPAC =DISCOEF(N,IND1,NS,IMGT)*frac_struc_cap(n,ind1,
     $        month) * (STAGEM - TW) ** PWR(N,IND1,NS,IMGT)
           ELSE
              CAPAC = 0.0
           ENDIF
        ENDIF
C
        IF ((IPURPSE(N,IND1,NS) .EQ. 2 .OR. IPURPSE(N,IND1,NS) .EQ. 3)
     $  .AND. IDNCLIDX .NE. M) THEN
           CAPACFLF(IMGT) = CAPAC
        ELSE
           CAPACFLF(IMGT) = 0.0
        ENDIF
        IF (IPURPSE(N,IND1,NS) .EQ. 1 .OR. IPURPSE(N,IND1,NS) .EQ. 3)
     $  THEN
           CAPACWSF(IMGT) = CAPAC + CAPACWSF(IMGT)
        ENDIF
C
        IF ( ISPEC(N,IND1) .EQ. 0) THEN
           IF( LOGDISGF) THEN
C
              IF (IRATING(N,IND1,NS,IMGT) .EQ. 1) THEN
                 IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
                    COUTFLF = CAPACFLF(IMGT)
                 ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                    PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,
     $              IND1 ,ISEAS))** 3* (STAGEM - RMNSTGS(N,IND1,ISEAS))
     $              **3
                    COUTFLF = PCTGO * CAPACFLF(IMGT)
                 ELSE
                    COUTFLF = 0.0
                 ENDIF
              ELSE
                 COUTFLF = CAPACFLF(IMGT)
              ENDIF
```

```
C
            ELSE
               COUTFLF = 0.0
            ENDIF
            IF (flnm(iflo).eq.'S335 ' .and. nint(s337fc).gt.0) COUTFLF
     &        = min(COUTFLF,s337fc)
            IF (.NOT. LFLWTRG(ids_trg_idx(N,IND1))) THEN
               call match (flnm(iflo),1,6,struc_name_w_targ ,
     $         no_of_struc_w_ds_targ_no_res,6,istruc_targ_indx)
               COUTFLF = COUTFLF * frc_flow_targ(istruc_targ_indx)
            ENDIF
C
            STROUT2 = COUTFLF + STROUT2
            IF (FLNM(IFLO) .EQ. strname_spec(22))COUTFL332 = COUTFLF
            IF (FLNM(IFLO) .EQ. strname_spec(23))COUTFL18C = COUTFLF
         ENDIF
   20 CONTINUE
      IF (ISPEC(N,IND1) .EQ. 0) THEN
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF ( FLNM(IFLO) .EQ. strname_spec(22) .OR. FLNM(IFLO) .EQ.
     $   strname_spec(23)) THEN
C
            IF (FLNM(IFLO) .EQ. strname_spec(22)) THEN
               IF (opt_flow_to_ts .EQ. 'TSMINDL') THEN
                  WSDMND = VOL332(MONTH) - COUTFL332
               ELSE
                  WSDMND = VOL332DO(MONTH) - COUTFL332
               ENDIF
               WSDMND = AMAX1(WSDMND,0.0)
               CAPCWS = amax1(DISCOEF(N,IND1,1,1)-COUTFL332,0.0)
               COUTWS = AMIN1(WSDMND,CAPCWS)
            ELSE
               WSDMND = AMAX1(VOL18C(MONTH)/86400.,DQU(IDNCLIDX)) -
     $         COUTFL18C
               WSDMND = AMAX1(WSDMND,0.0)
               CAPCWS = amax1(DISCOEF(N,IND1,1,1)-COUTFL18C,0.0)
               COUTWS = AMIN1(WSDMND,CAPCWS)
            ENDIF
         ELSE
            WSDMND = amax1(tdqu_dn(IDNCLIDX) - COUTFLF,0.0)
            rem_capacws = amax1(CAPACWSF(IMGT) - COUTFLF,0.0)
            COUTWS = AMIN1(rem_capacws,WSDMND)
C           if (iactual_flow_sim .eq. 1) print *,COUTWS,COUTFLF
C    $      ,rem_capacws,CAPACWSF(IMGT),upstream_inflow_lim,' lll'
         ENDIF
C
         STROUT2 = STROUT2 + COUTWS
C
C  CALCULATE MAXIMUM AVAILABLE STORAGE IN RESERVOIR RECEIVING DISCHARGE
C   FROM STRUCTURE,IF APPLICABLE
C
         DIFFDRS = 0.0
         COUTRS = STROUT2
         TCOUTRS = 0.0
         IF (NRES(N,IND1) .GT. 0) THEN
            DO IR = 1,NRES(N,IND1)
```

```
               COUTTP = PCTRES(N,IND1,IR) * STROUT2 + DIFFDRS
               COUTRS = AMIN1(COUTTP,VOLWTR(IRSRCVR(N,IND1,IR)))
               DIFFDRS = STROUT2 * PCTRES(N,IND1,IR) - COUTRS
               TCOUTRS = TCOUTRS + COUTRS
            END DO
            DO IR = 1,NRES(N,IND1)
               IF (PCTRES(N,IND1,IR) .LT. 0.010) THEN
                  icres_dn = 0
                  DO j = 1,nstrctr(idnclidx)
                     IF (nres(idnclidx,j) .GT. 0) THEN
                        DO irr = 1,nres(idnclidx,j)
                           IF (irsrcvr(idnclidx,j,irr) .EQ. IRSRCVR
     $                     (N,IND1,IR)) THEN
                              icres_dn = icres_dn + 1
                           ENDIF
                        END DO
                     ENDIF
                  END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                  IF (icres_dn .NE. 0) THEN
                     IF (resname(IRSRCVR(N,IND1,IR)) .EQ.
     $               name_res_spec_code(15)) THEN
                        temp_availstor = tempf_volwtr(IRSRCVR(N,IND1,IR)
     $                  ) - outf_from_c11res
                        temp_availstor = amax1(temp_availstor,0.0)
                        temp_struccap = tempf_weircff(IRSRCVR
     $                  (N,IND1,IR) ,i_res_inf_struc_indx(n,ind1,ir)) -
     $                  outf_from_c11res
                     ELSE
                        temp_availstor = tempf_volwtr(IRSRCVR
     $                  (N,IND1,IR))
                        temp_struccap = tempf_weircff(IRSRCVR
     $                  (N,IND1,IR) ,i_res_inf_struc_indx(n,ind1,ir))
                     ENDIF
                     strout2=amin1(strout2,temp_availstor,temp_struccap)
                     tempf_volwtr(IRSRCVR(N,IND1,IR)) = tempf_volwtr
     $               (IRSRCVR(N,IND1,IR)) - STROUT2/NRES(N,IND1)

                     tempf_weircff(IRSRCVR(N,IND1,IR),
     $               i_res_inf_struc_indx (n,ind1,ir)) = tempf_weircff
     $               (IRSRCVR(N,IND1,IR) ,i_res_inf_struc_indx
     $               (n,ind1,ir)) - STROUT2/NRES(N,IND1)
                  ELSE
                     IF (resname(IRSRCVR(N,IND1,IR)) .EQ.
     $               name_res_spec_code(15)) THEN
                        temp_availstor = volwtr(IRSRCVR(N,IND1,IR)) -
     $                  outf_from_c11res
                        temp_availstor = amax1(temp_availstor,0.0)
                     ELSE
                        temp_availstor = volwtr(IRSRCVR(N,IND1,IR))
                     ENDIF
                     strout2 = amin1(strout2,temp_availstor)
                  ENDIF
               ELSE
                  strout2 = TCOUTRS
               ENDIF
            END DO
```

```
         ENDIF
      ELSE
C
C-----------------------------------------------------------------
C          GAPS IN L-5 TO TRANSFER WATER FROM HOLEY LAND TO WCA3
C-----------------------------------------------------------------
C
         GWDPTH = (ELLS(NODEHL) - H(NODEHL)) * S(NODEHL)
         IF (POND(NODEHL) .GT. GWDPTH) THEN
            HLSTAGE = POND(NODEHL) - GWDPTH + ELLS(NODEHL)
         ELSE
            HLSTAGE = (POND(NODEHL) - GWDPTH)/S(NODEHL) + ELLS(NODEHL)
         ENDIF
         call bpts(iday,month,10.0,7,1,1,holy_sched)
         IF (FLNM(IFLO) .EQ. strname_spec(26) .OR. FLNM(IFLO) .EQ.
     $   strname_spec(27) .OR. FLNM(IFLO) .EQ. strname_spec(28)) THEN
            IF (hlyenv .EQ. 'TRUE ') THEN
               holy_stage = stage_import_wca(iholy)
            ELSE
               holy_stage = holy_sched
            ENDIF
            IF (.NOT. DRYSEAS .OR. HLSTAGE .GE. holy_stage + 1.0) THEN

C
               STROUT2 = CAPACFLF(2)
            ELSE
               STROUT2 = CAPACFLF(1)
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C                   PUMP FROM C51W TO PROP L8 BPIT RES
C-----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(30)) THEN
            STROUT2 = MIN(tot_avl_l8_bpit_res
     $      ,DISCOEF(N,IND1,1,1)-rl8runoff_to_bptres
     $      ,VOLWTR(irsrcvr(n,ind1,1)))

C-----------------------------------------------------------------
C                            L31N PUMP STATION
C-----------------------------------------------------------------
C
         ELSEIF (char_struc_name(1:5) .eq. 'S356 ') THEN
            IF (KFLO(IFLS334) .NE. 0 .OR. DQU(L31S) .GT. 0.0) THEN
               CAPACFLF(ISEAS) = 0.0
            ELSE
               CAPACFLF(ISEAS) = DISCOEF(N,IND1,1,1)
     &                  * frac_struc_cap(n,ind1,month)
            ENDIF
              DIFFS = RMXSTG(N,IND1,ISEAS)-RMNSTGS(N,IND1,ISEAS)
              IF (STAGEM .LT. RMNSTGS(N,IND1,ISEAS)+DIFFS .AND.
     &         STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                QPUMP = (STAGEM - RMNSTGS(N,IND1,ISEAS)) *(1.0/DIFFS)
     &            * DISCOEF(N,IND1,1,1) * frac_struc_cap(n,ind1,month)
              ELSEIF (STAGEM .GE. RMNSTGS(N,IND1,ISEAS)+DIFFS) THEN
                QPUMP = DISCOEF(N,IND1,1,1)
     &            *frac_struc_cap(n,ind1,month)
              ELSE
```

```
            QPUMP = 0.0
          ENDIF
        AVVOL_UP = (STAGEM-RMNSTGS(N,IND1,ISEAS))*CL(N)*WIDTH(N) +
$         CGSEEP(N,RMNSTGS(N,IND1,ISEAS),0,0.)+TOVFLC(N,
$          RMNSTGS(N,IND1,ISEAS),0,0.,ITER,1) + QU(N)
        AVVOL_UP = AMAX1(AVVOL_UP,0.0)/86400.
        if (iseep_lim_s356_opt .ne. 0) then
           AVVOL_UP_S = AMAX1(QU(N) - KFLO(IFLS335)*86400.
$            - KFLO(IFLS334)*86400.- s334_fc*86400.,0.0)
           AVVOL_UP = amin1(AVVOL_UP_S/86400.,AVVOL_UP)
        endif
        QPUMPT =CAPACFLF(ISEAS)
        STROUT2 =AMIN1(QPUMPT,DISCOEF(N,IND1,1,1),AVVOL_UP,QPUMP)
        if (irceive(n,ind1) .eq. 1) then
         if ( opt_mrp_to_enp) then
          if (chdep(IDNCLIDX) .gt. CNLSTG(N,IND1,1)-0.25) then
            frac_max_disc_c = (CNLSTG(N,IND1,1)-chdep(idnclidx))
$            / 0.25
            frac_max_disc_c = amax1(frac_max_disc_c,0.0)
          else
            frac_max_disc_c = 1.0
          endif
         else
          frac_max_disc_c = 1.0
         endif
        else
          frac_max_disc_c = 1.0
          if (stage3b .gt. 8.5 .or. nint(rmnstg_dual_ops0(n,ind1))
$             .eq. 99) then
            strout2 = 0.0
          endif
        endif
        DPTHGW = (ELLS(NODE_G3273) - H(NODE_G3273)) * S(NODE_G3273)
        IF (POND(NODE_G3273) .GT. DPTHGW) THEN
          STAGE_G3273 = POND(NODE_G3273) - DPTHGW + ELLS(NODE_G3273)
        ELSE
          STAGE_G3273 = (POND(NODE_G3273)-DPTHGW)/S(NODE_G3273)
$                              + ELLS(NODE_G3273)
        ENDIF
        if (opt_mrp_to_enp .and. irceive(n,ind1) .eq. 1) then
          DO ncal = 1,nch
             DO i=1,nstrctr(ncal)
               IF (tmpname(ncal,i).eq.'S333  ') THEN
                 istruc_s333=i
                 incl_s333 = ncal
               ENDIF
             END DO
          ENDDO
          if (STAGE_G3273 .gt. stg_trigger_fl(incl_s333,istruc_s333
$          ,1)) then
            frac_max_disc_gc = 0.0
          else
            frac_max_disc_gc = 1.0
          endif
        else
          frac_max_disc_gc = 1.0
        endif
```

```
            STROUT2 = STROUT2 * amin1(frac_max_disc_gc,frac_max_disc_c)
C-----------------------------------------------------
C                       S356A
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(32)) THEN
            IF (KFLO(IFLS334) .NE. 0 .OR. nint(DQU(L31S)) .GT. 0 .OR.
     $      nint(DQU(IC4)) .GT. 0) THEN
               CAPACFLF(ISEAS) = 0.0
            ELSE
               CAPACFLF(ISEAS) = DISCOEF(N,IND1,1,1)
            ENDIF
            QPUMPT =CAPACFLF(ISEAS)
            STROUT2   =AMIN1(QPUMPT,DISCOEF(N,IND1,1,1))
C
C-------------------------------------------------------------
C             WATER SUPPLY FROM L-8 TO M-CANAL
C-------------------------------------------------------------
C
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(33)) THEN
            HDIFF = ABS(STAGEM - CHDEP(IDNCLIDX))
            IF (STAGEM .LT. CHDEP(IDNCLIDX)) THEN
               IF (STAGEM .GT. 10.2) THEN
                  CAPAC = 7. 15* (6.0 - HDIFF) + 182.0
               ELSEIF (STAGEM .GE. 9.5) THEN
                  IF (HDIFF .GT. 11.5) THEN
                     CAPAC = 136.0 - 10.29* (HDIFF - 11.5)
                  ELSEIF (HDIFF .GT. 4.0) THEN
                     CAPAC = 136.0 + 4.53 * (11.5 - HDIFF)
                  ELSE
                     CAPAC = 170.0
                  ENDIF
               ELSEIF (STAGEM .GT. 9.1) THEN
                  CAPAC = 150.0* (STAGEM -9.1) / 0.4
               ELSE
                  CAPAC = 0.0
               ENDIF
            ELSE
               IF (STAGEM .GE. 9.5) THEN
                  CAPAC = 170.0
               ELSEIF (STAGEM .GT. 9.1) THEN
                  CAPAC = 170.0 * (STAGEM - 9.1) / 0.4
               ELSE
                  CAPAC = 0.0
               ENDIF
            ENDIF
            CAPAC = CAPAC*FACTS2MC
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            GWDPTH = (ELLS(IWPBCAT)-H(IWPBCAT))*S(IWPBCAT)
            IF (POND(IWPBCAT) .GT. GWDPTH) THEN
               STAGE = POND(IWPBCAT) - GWDPTH + ELLS(IWPBCAT)
            ELSE
               STAGE = (POND(IWPBCAT) - GWDPTH)/S(IWPBCAT) + ELLS
     $         (IWPBCAT)
            ENDIF
C
            IF (STAGEM .GT. rmin_l8stg_flow_to_mcnl(iseas)) THEN
```

```
                IF (STAGE .LT. rmax_wpbcat_for_inflw) THEN
                   stg_rmx = rmax_wpbcat_for_inflw
                   DISHGDN = AMIN1(DQU(ILKMG),50.0)
                   AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $             (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $             (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN
     $             *86400.
                   AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
                   capac_out = capac
C        IF (stagem .GT. rmin_l8stg_flow_to_mcnl(iseas) + 0.25) THEN
C          capac_out = capac
C        ELSE
C          capac_out = 4.0*capac*(stagem-rmin_l8stg_flow_to_mcnl(iseas))
C        ENDIF
                   STROUT2 = AMIN1(CAPAC_OUT,AVVOLDN)
                   IF (STAGELO.LT.SCHED(IDAY,MONTH)) STROUT=AMIN1(CAPAC ,
     $             DQU(ILKMG)+DMNDAGMC,50.0)
                ELSE
                   STROUT2 = 0.0
                ENDIF
             ELSE
                stg_rmx = CRMIN(IDNCLIDX,iseas)
                DISHGDN = AMIN1(DQU(ILKMG),50.0)
                AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $          (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $          (IDNCLIDX,stg_rmx, 0,0.,iter,2) - QINFLOWN(N) + DISHGDN
     $          *86400.
                AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
                STROUT2 = AMIN1(CAPAC,AVVOLDN)
                IF (STAGELO.LT.SCHED(IDAY,MONTH)) STROUT2=AMIN1(CAPAC,DQU
     $          (ILKMG)+DMNDAGMC,50.0)
             ENDIF
C-------------------------------------------------------
C        PUMP TO C9RES FROM US27S
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(83)) THEN
C

             rem_capac = amax1(capacflf(imgt) - outf_to_c9res_via_us27s,
     $       0.0)
             rem_pump_cap = DISCOEF(n,ind1,1,1) - outf_to_c9res_via_us27s
             rem_avvol = VOLWTR(IRSRCVR(N,IND1,1))
             strout2 = amin1(rem_avvol,rem_capac,rem_pump_cap)
C-------------------------------------------------------
C            S-9
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(85)) THEN
C
             strout2 = capacflf(imgt)
C-------------------------------------------------------
C            S-9A
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(86)) THEN
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             strout2 = discoef(n,ind1,1,1)
C-----------------------------------------------------------
```

```
C          PROPSED STRUC IN L-37 Canal to PASS ENVIRONMENTAL FLOWS
C-------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(59)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))** 3* (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               strout2 = amax1(coutfl,prevqu(n)/86400.)
               strout2 = amin1(cout,CAPACFLF(IMGT))
            ELSE
               strout2 = coutfl
            ENDIF
C-------------------------------------------------------------
C          PROPSED STRUC IN L-30 to PASS ENVIRONMENTAL FLOWS
C-------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(61)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))** 3* (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               strout2 = amax1(coutfl,prevqu(n)/86400.)
               strout2 = amin1(cout,CAPACFLF(IMGT))
            ELSE
               strout2 = coutfl
            ENDIF
C-------------------------------------------------
C              S-335
C-------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(6)) THEN
            IF (STAGEM .GT. RMXSTG(N,IND1,ISEAS)) THEN
               COUTFL = CAPACFLF(IMGT)
            ELSEIF(STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               PCTGO = 1.0/(RMXSTG(N,IND1,ISEAS) - RMNSTGS(N,IND1 ,
     $         ISEAS))**3 * (STAGEM - RMNSTGS(N,IND1,ISEAS))**3
               COUTFL = PCTGO * CAPACFLF(IMGT)
            ELSE
               COUTFL = 0.0
            ENDIF
            IF (LWSLO(ig1502)) THEN
               struc_inflw = amax1(prevqu(n)/86400.-tseepge(ilv30),0.0)
               strout2 = amax1(coutfl,struc_inflw)
               strout2 = amin1(strout2,CAPACFLF(IMGT))
            ELSE
               strout2 = coutfl
            ENDIF
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------
C        M1 LOWER BASIN TO MCNL
C-----------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(34)) THEN
C
C  downstream avail storage
            DISHGDN = 0.0
            TOT_PUMP = 0.0
            stg_rmx = rmax_wpbcat_for_inflw
            DISHGDN = AMIN1(DQU(ILKMG),50.0)
            AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $      (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $      (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN*86400.
            AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
C
C  calculate outflow
            STROUT2 = amin1(avvoldn,DISCOEF(N,IND1,1,1))
C-----------------------------------------
C        M1 LOWER BASIN TO MCNL
C-----------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(35)) THEN
C
C  downstream avail storage
            DISHGDN = 0.0
            TOT_PUMP = 0.0
            stg_rmx = rmax_wpbcat_for_inflw
            DISHGDN = AMIN1(DQU(ILKMG),50.0)
            AVVOLDN =(stg_rmx-CHDEP(IDNCLIDX))*CL(IDNCLIDX)*WIDTH
     $      (IDNCLIDX) - CGSEEP(IDNCLIDX,stg_rmx,0,0.) - TOVFLC
     $      (IDNCLIDX,stg_rmx, 0,0.,iter,2)-QINFLOWN(N) + DISHGDN*86400.
            AVVOLDN = AMAX1(AVVOLDN,0.0)/86400.
C
C  calculate outflow
            STROUT2 = amin1(avvoldn,DISCOEF(N,IND1,1,1))
C
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(70)) THEN
            IF (stagem .LT. crel(n,1)) THEN
               IF (stagem .GT. RMNSTGS(N,IND1,ISEAS)+0.5) THEN
                  strout2 = capacflf(imgt)
                  IF (stagem .GT. RMNSTGS(N,IND1,ISEAS)) THEN
                     strout2 = 2.0* capacflf(imgt)*(stagem - RMNSTGS(N,
     $               IND1,ISEAS))
                  ELSE
                     strout2 = 0.0
                  ENDIF
               ELSE
                  strout2 = 0.0
               ENDIF
            ELSE
               strout2 = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C              S142 BACK TO WCA3A
C-----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(39)) THEN
```

```
            IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
               S142CAP = DISCOEF(N,IND1,1,1)*(STAGEM-CHDEP(IDNCLIDX))
     $         **PWR(N,IND1,1,1)
            ELSE
               S142CAP = 0.0
            ENDIF
            STROUT2 = AMIN1(S142CAP,TFLG123)
C
C-----------------------------------------------------
C   PROP PUMP DIVERTING C11W OUTFLOW TO SOUTH
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(42)) THEN
            CAPAC = DISCOEF(N,IND1,1,1)
            strout2 = capac
C-----------------------------------------------------
C                 S-174
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(64)) THEN
            IF (IRATING(N,IND1,1,IMGT) .NE. 2) THEN
               IF (DELTAH .GT. RMINHD(N,IND1)) THEN
                  S174CAP = DISCOEF(N,IND1,1,IMGT)*(DELTAH - RMINHD
     $            (N,IND1)) ** PWR(N,IND1,1,IMGT)
               ELSE
                  S174CAP = 0.0
               ENDIF
            ELSE
               S174CAP = DISCOEF(N,IND1,1,IMGT)
            ENDIF
            s174cap = amin1(s174cap,STRDCAP(N,IND1,1,IMGT))
            IF (opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               IF (dqu(idnclidx) .GT. 0.0 .OR. LWSLO(itylsl)) THEN
                  IF (LWSLO(itylsl)) THEN
                     STROUT2 = s174cap
                  ELSE
                     STROUT2 = amin1(s174cap,dqu(idnclidx))
                  ENDIF
               ELSE
                  STROUT2 = 0.0
               ENDIF
            ELSE
               STROUT2 = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-------------------------------------------------------
C                 S-175
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(65)) THEN
            deltah = stagem - (chdep(idnclidx)+hdc(idnclidx,iseas))
            IF (DELTAH .GT. RMINHD(N,IND1)) THEN
               S175CAP = DISCOEF(N,IND1,1,IMGT) * (DELTAH - RMINHD
     $         (N,IND1)) ** PWR(N,IND1,1,IMGT)
            ELSE
               S175CAP = 0.0
            ENDIF
            IF ( opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               IF (main_pres_level_fl_prot_ts .EQ. 'FALSE') THEN
                  IF (.NOT. LWSLO(itylsl)) THEN
```

```
                     stg_min = stage_import_wca(il31w_min_rf) + 0.1
                     stg_max = stage_import_wca(il31w_min_rf) + 0.3
                  ELSE
                     stg_min = stage_import_wca(il31w_min_rf) + 0.3
                     stg_max = stage_import_wca(il31w_min_rf) + 0.7
                  ENDIF
               ELSE
                  stg_max = RMXSTG(N,IND1,ISEAS)
                  stg_min = RMNSTGS(N,IND1,ISEAS)
               ENDIF
               IF (STAGEM .GT. stg_max) THEN
                  STROUT2 = s175cap
               ELSE IF (STAGEM .GT. stg_min) THEN
                  STROUT2 = 1.0/(stg_max-stg_min)*s175cap*(STAGEM
     $            -stg_min)
               ELSE
                  STROUT2 = 0.0
               ENDIF
            ELSE
               STROUT2 = 0.0
            ENDIF
C-----------------------------------------------------
C               S-332
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(22)) THEN
            IF (s332sparrow_opt .EQ. 'TRUE ') THEN
               IF (month.EQ. 7 .AND. iday .GT. 15) THEN
                  frac_struc_cap_s332 = frac_struc_cap(n,ind1,month+1)
               ELSE
                  frac_struc_cap_s332 = frac_struc_cap(n,ind1,month)
               ENDIF
            ELSE
               frac_struc_cap_s332 = frac_struc_cap(n,ind1,month)
            ENDIF
            s332cap = DISCOEF(n,ind1,1,1)*frac_struc_cap_s332
            IF (opt_flow_to_ts .EQ. 'TSRFPLN') THEN
               STROUT2 = s332cap
            ELSE
               STROUT2 = 0.0
            ENDIF
C-------------------------------------------------------
C                          S-197
C-------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(68)) THEN
            IF (stagem .GT. rmnstgs(n,ind1,iseas)) THEN
               frac_go = (stagem-rmnstgs(n,ind1,iseas))** 2/ (rmxstg(n,
     $         ind1,iseas)-rmnstgs(n,ind1,iseas))**2
               frac_go = amin1(frac_go,1.0)
            ELSE
               frac_go = 0.0
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            tw197 = get_tide(ycnlnode(n,1),jday,proj_sea_level_rise)
            tw = 1.025*(tw197+2.0)-2.0
            IF (stagem .GT. tw) THEN
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
```

```
C          IF (conv .EQ. 1)
C    +     PRINT *,icnl_no_fc(n,ind1,ic),stg_trig_oper(n,ind1,ic,1)
                 IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $           (n,ind1,ic,1)) THEN
                    frac_full = 7.0/13.0 + ((min(chdep(icnl_no_fc
     $              (n,ind1,ic)) - stg_trig_oper(n,ind1,ic,1),0.1))**2
     $              /(0.1)**2)*(6.0/13.0)
                    cout = min(frac_full,frac_go) * DISCOEF
     $              (N,IND1,1,1) * (stagem - TW) ** PWR(N,IND1,1,1)
     $              * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,1),cout,chdep(n),'  AAA'
                    GO TO 30
                 ENDIF
               END DO
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
                 IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $           (n,ind1,ic,2)) THEN
                    frac_full = 3.0/13.0 + ((chdep(icnl_no_fc
     $              (n,ind1,ic)) - stg_trig_oper(n,ind1,ic,2))**2/(0.5
     $              *(stg_trig_oper(n ,ind1,ic,1)-stg_trig_oper(n,ind1,
     $              ic,2))**2))*(4.0/13.0)
                    frac_full = min(frac_full,7.0/13.0)
                    cout = min(frac_full,frac_go)*DISCOEF(N,IND1,1,1)
     $              * (stagem - TW) ** PWR(N,IND1,1,1)
     $              * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,2),cout,chdep(n),'  BBB'
                    GO TO 30
                 ENDIF
               END DO
               DO ic = 1,n_trigger_fc_cnl(n,ind1)
                 IF (chdep(icnl_no_fc(n,ind1,ic)) .GT. stg_trig_oper
     $           (n,ind1,ic,3)) THEN
                    frac_full = ((chdep(icnl_no_fc(n,ind1,ic)) -
     $              stg_trig_oper(n,ind1,ic,3))**2/(0.5*(stg_trig_oper
     $              (n ,ind1,ic,2)-stg_trig_oper(n,ind1,ic,3))**2))
     $              *(3.0/13.0)
                    frac_full = min(frac_full,3.0/13.0)
                    cout = amin1(frac_full,frac_go)*DISCOEF(N,IND1,1,1)
     $              * (stagem - TW) ** PWR(N,IND1,1,1)
     $              * frac_struc_cap(N,IND1,month)
C            IF (conv.EQ. 1)
C    +       PRINT *,icnl_no_fc(n,ind1,ic),chdep(icnl_no_fc(n,ind1,ic))
C    +         ,stg_trig_oper(n,ind1,ic,3),cout,chdep(n),'  CCC'
                    GO TO 30
                 ENDIF
               END DO
               cout_mx = frac_go * DISCOEF(N,IND1,1,1) * (stagem - TW)
     $         ** PWR(N,IND1,1,1) * frac_struc_cap(N,IND1,month)
               IF (nint(dqu(lc111)) .EQ. 0) THEN
                  cout = amin1(cout_mx,kflo(ifls176))
               ELSE
                  cout = 0.0
               ENDIF
```

```
            ELSE
               cout = 0.0
            ENDIF
   30       strout2 = cout
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------
C                 S-331 AND S173
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(67)) THEN
            DPTHGW = (ELLS(NODANGL) - H(NODANGL)) * S(NODANGL)
            IF (POND(NODANGL) .GT. DPTHGW) THEN
               STG_ANGELS1 = POND(NODANGL) - DPTHGW + ELLS(NODANGL)
            ELSE
               STG_ANGELS1 = (POND(NODANGL)-DPTHGW)/S(NODANGL) + ELLS
     $         (NODANGL)
            ENDIF
            DPTHGW = (ELLS(IG2517) - H(IG2517)) * S(IG2517)
            IF (POND(IG2517) .GT. DPTHGW) THEN
               STG_ANGELS2 = POND(IG2517) - DPTHGW + ELLS(IG2517)
            ELSE
               STG_ANGELS2 = (POND(IG2517)-DPTHGW)/S(IG2517) + ELLS
     $         (IG2517)
            ENDIF
            stg_angels = (stg_angels1 + stg_angels2)/2.0
            LANGELX = stg_angels .GE. stg_trigger_fl(n,ind1,3)
            IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
               str_cap(1) = DISCOEF(N,IND1,1,1) * (STAGEM-CHDEP
     $         (IDNCLIDX)) * PWR(N,IND1,1,1)
            ELSE
               str_cap(1) = 0.0
            ENDIF
C
            LANGEL_MXX = .NOT.(stg_angels .LT. stg_trigger_fl(n,ind1,2))
            IF (stg_angels .GT. stg_trigger_fl(n,ind1,1).OR. LANGEL_MXX)
     $      THEN
               LANGEL_MXX = .TRUE.
               stg_min = stg_min_fc(n,ind1,2)
            ELSE IF (stg_angels .GT. stg_trigger_fl(n,ind1,3)) THEN
               stg_min = stg_min_fc(n,ind1,1)
            ELSE
               IF (runmode .EQ. 'CALIB') THEN
                  stg_min = 5.0
               ELSE
                  stg_min = 5.2
               ENDIF
            ENDIF
            idrawdwnc = 0
            do ll = 1,idrwn
              if (n .eq. icanal_no_drawdown(ll)) then
                idrawdwnc = 1
              endif
            enddo
            if (nstorm_day(ndays_sim_sd).eq.1 .and. idrawdwnc.eq.1) then
               stg_min = stg_min - 1.0
            endif
            idualopsc = 0
            do ll = 1,ndual_canal_ops(1)
```

```
              if (n .eq. icanal_no_dualops(ll,1)) then
                 idualopsc = 1
              endif
            enddo
            if (nstorm_day(ndays_sim_sd).ne.1 .and. idualopsc.eq.1) then
               stg_min = stg_min - 0.5
            endif
C
            COUT_FC = 0.0
            DO ns = 1,nsubstr(n,ind1)-1
               IF (ipurpse(n,ind1,ns) .NE. 1) THEN
                  IF( LOGDISGF) THEN
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                     IF (LANGELX) THEN
                        IF (STAGEM .GT. stg_min + 0.4) THEN
                           COUTFLF = str_cap(ns)
                        ELSEIF(STAGEM .GT. stg_min) THEN
                           PCTGO = 6.25* (STAGEM - stg_min)**2
                           COUTFLF = PCTGO * str_cap(ns)
                        ELSE
                           COUTFLF = 0.0
                        ENDIF
C              ELSE
                        IF (STAGEM .GT. 6.2) THEN
                           COUTFLF = str_cap(ns)
                        ELSEIF(STAGEM .GT. 6.0) THEN
                           PCTGO = 25.0* (STAGEM - 6.0)**2
                           COUTFLF = PCTGO * str_cap(ns)
                        ELSE
                           COUTFLF = 0.0
                        ENDIF
                     ENDIF
                  ELSE
                     COUTFLF = 0.0
                  ENDIF
               ELSE
                  COUTFLF = 0.0
               ENDIF
               COUT_FC = COUT_FC + COUTFLF
            END DO
            ilast_subs = nsubstr(n,ind1)
            IF (INGVHDOPT(N,IND1,ilast_subs) .EQ. 1) THEN
               IF (STAGEM .LE. TW) THEN
                  CAPACP = DISCOEF(N,IND1,ilast_subs,IMGT)
     $             * frac_struc_cap(N,IND1,month)
               ELSE
                  CAPACP = 0.0
               ENDIF
            ELSE
               CAPACP = DISCOEF(N,IND1,ilast_subs,IMGT)
     $             * frac_struc_cap(N,IND1,month)
            ENDIF
            IF (ipurpse(n,ind1,ilast_subs) .NE. 1) THEN
               IF (LOGDISGF) THEN
                  CAPAC = CAPACP
               ELSE
```

```
                 CAPAC = 0.0
              ENDIF
           ELSE
              CAPAC = 0.0
           ENDIF
           str_cap(ilast_subs) = CAPAC
           STROUT2 = cout_fc + str_cap(ilast_subs)
C-------------------------------------------------------------
C  PROPOSED PUMP : C-18 TO LOX SLOUGH
C-----------------------------------------------
        ELSE IF (flnm(iflo) .EQ. 'LXSLWS') THEN
           IF (loxenv .EQ. 'FALSE') THEN
              strout2 = 0.0
           ELSE
              IF (LWSLO(iloxah)) THEN
                 strout2 = DISCOEF(N,IND1,1,1)
              ELSE
                 strout2 = 0.0
              ENDIF
           ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------------------
C             WATER SUPPLY FROM CATCHMENT AREA TO  EASTERN
C               REACH OF M - CANAL
C-------------------------------------------------------------
        ELSEIF (FLNM(IFLO) .EQ. strname_spec(38)) THEN
           IF (STAGEM .GT. 14.5) THEN
              RMXDSCHG = DISCOEF(N,IND1,1,1)
           ELSEIF(STAGEM .GT. 12.5) THEN
              RMXDSCHG = 12.5* (STAGEM-12.5) + 25.0
           ELSE
              RMXDSCHG = 25.0
           ENDIF
           STROUT2 = AMIN1(DQU(ILKMG),RMXDSCHG)
C-----------------------------------------------
C             BACKFLOW FROM L-8 INTO LAKE OKEECHOBEE
C-----------------------------------------------------------
        ELSEIF (FLNM(IFLO) .EQ. strname_spec(37)) THEN
C
           IF (STAGEM+HDC(N,iseas) .GT. STAGELO) THEN
C
              STROUT2 = CAPACFLF(1)
           ELSE
              STROUT2 = 0.0
           ENDIF
C-------------------------------------------------------------
C                STRUCTURE TO ALLOW BACK FLOW BY
C                 GRAVITY FROM M-CANAL TO L-8
C-------------------------------------------------------------
        ELSEIF (FLNM(IFLO) .EQ. strname_spec(36)) THEN
           rmin_stg_fc = rmnstgs(n,ind1,iseas)
           IF (STAGEM+HDC(N,iseas) .GT. rmin_stg_fc + 1.0) THEN
              STROUT2 = DISCOEF(N,IND1,1,1)
           ELSEIF (STAGEM+HDC(N,iseas) .GT. rmin_stg_fc) THEN
              STROUT2 = 1. 0* DISCOEF(N,IND1,1,1) * (STAGEM + HDC
     $        (N,iseas) - rmin_stg_fc)
```

```
            ELSE
               STROUT2 = 0.0
            ENDIF
C
C-----------------------------------------------------------------
C             WATER SUPPLY FROM EASTERN M-CANAL TO CLEAR LAKE
C----------------------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(54)) THEN
            IF (STAGEM .GT. RMNSTGS(N,IND1,ISEAS)) THEN
               COUTWS = DQU(ILKMG)
               STROUT2 = COUTWS
            ELSE
               IF (STAGEM .GT. CHDEP(IDNCLIDX)) THEN
                  CAPACITY = DISCOEF(N,IND1,1,1)*(STAGEM-CHDEP(IDNCLIDX)
     $            ) ** PWR(N,IND1,1,1)
               ELSE
                  CAPACITY = 0.0
               ENDIF
               STROUT2 = AMIN1(DQU(ILKMG),CAPACITY)
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------------------------------------------
C                         S-316
C                      PROPOSED STRCTR
C-----------------------------------------------------
         ELSEIF (FLNM(IFLO) .EQ. strname_spec(69)) THEN
            IF (DQU(IDNCLIDX) .GT. 0.0) THEN
               STGDN = CHDEP(IDNCLIDX)+0.1
               AVVOL_UP = (STAGEM-STGDN) *CL(N)*WIDTH(N) + CGSEEP(N,
     $         STGDN,0,0.)+TOVFLC(N,STGDN,0,0.,iter,2)
               AVVOL_UP = AMAX1(AVVOL_UP/86400.,0.0)
               STROUT2 =AMIN1(AVVOL_UP,DISCOEF(N,IND1,1,1),DQU(IDNCLIDX)
     $         )
            ELSE
               AVVOL_UP = (STAGEM-RMNSTGS(N,IND1,ISEAS))*CL(N)*WIDTH (N)
     $         +CGSEEP(N,RMNSTGS(N,IND1,ISEAS),0,0.)+TOVFLC(N, RMNSTGS
     $         (N,IND1,ISEAS),0,0.,iter,2)
               AVVOL_UP = AMAX1(AVVOL_UP/86400.,0.0)
               STROUT2 = AMIN1(AVVOL_UP,DISCOEF(N,IND1,1,1))
            ENDIF
C
C-----------------------------------------------------------------
C                S-319 PROPOSED PUMP FOR C51W
C-----------------------------------------------------------------
         ELSEIF(FLNM(IFLO) .EQ. strname_spec(55)) THEN
            out_flow_east = 0.0
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
               capac_s155a = 625.0* (stagem-(chdep(ic51) +hdc
     $         (ic51,iseas))) ** 0.5
            ELSE
               capac_s155a = 0.0
            ENDIF
            cout = 0.0
            TDQU = AMAX1(DQU(IC51) - out_flow_east,0.0)
            WSDEL = TDQU
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
```

```
               COUT = COUT + amin1(WSDEL,capac_s155a)
            ENDIF
            out_flow_east = out_flow_east + amin1(WSDEL,capac_s155a)
            capac_s155a = capac_s155a - amin1(WSDEL,capac_s155a)
            COUT = WEIRCFF(5,1,1) + COUT
            IF (stagem .GT. chdep(ic51)+hdc(ic51,iseas)) THEN
               COUT = COUT + capac_s155a
               out_flow_east = out_flow_east + capac_s155a
               capac_s155a = 0.0
            ENDIF
            STROUT2 = COUT
         ELSE
            STROUT2 = 0.0
         ENDIF
      ENDIF
C
      RETURN
      END

C*****************************************************************************
C  This function calculates groundwater seepage to/from canal
C  to grid cell where canal passes through. The calculations
C  are performed for canal reaches.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION cgseep (n, stagec, ioptn, conv)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'smallres.inc'
      INCLUDE 'dlycslope.inc'

      REAL conv

      DIMENSION ilgtha(max_ncnlloc), nodeca(max_ncnlloc)
      DOUBLE PRECISION stagec

      NPTS1 = 0
      DSUM  = 0.0
      CGSEEP = 0.0
      DO 10 NP = 1,N-1
         NPTS1 = NPTS1 + NODCR(NP)
   10 CONTINUE
C
      DO 20 IC = 1,NODCR(N)
         GWSMX = 0.0
         NPTS1 = NPTS1 + 1
         NODECA(IC) = NODECN(NPTS1) / 10
         ILGTHA(IC) = MOD(NODECN(NPTS1) ,  10)
         NODE = NODECA(IC)
         DSUM = DSUM + DISTNC(ILGTHA(IC))
         HDIFF = HDC(N,iseas) * (CL(N) - DSUM) / CL(N)
         RCAR = DISTNC(ILGTHA(IC)) * WIDTH(N)
```

```
         SWL = STAGEC + HDIFF
         IF (SWL.LT.ELLS(NODE) .OR. H(NODE) .LT. ELLS(NODE) .OR.
     $    LCNB(N) .NE. IBSN(NODE)) THEN
            IF (LCNB(N) .NE. IBSN(NODE)) THEN
               GWSTAGE = H(NODE)+POND(NODE)
            ELSE
               GWSTAGE = H(NODE)
            ENDIF
            GWSMX = (SWL - GWSTAGE) * RCAR * CHHC(N) * DT * 1.6
            CGSEEP = CGSEEP - GWSMX
            IF (IOPTN .EQ. 1) THEN
               IF(CONV .EQ. 1.) RCHG(NODE) = RCHG(NODE) + GWSMX /(GDAR
     $         * (1.0-total_small_afact(node)-carfact(node)))
               IF(CONV .EQ. 1.) THEN
                  SEEP_TO_CANAL(NODE) = - GWSMX /43560.
                  rmonthly_seep_to_canal(node) = rmonthly_seep_to_canal
     $            (node) + SEEP_TO_CANAL(NODE)
               ENDIF
               IF (ivary_slope_daily_opt(n).eq.1) THEN
                  call match(cnm(N),1,5,canal_name_dvslope
     $            ,ncnl_dly_slope,5,imatch_slope)
                  IF (IC .LE. mid_point_canl(N)) THEN
                     dischg_for_slope(imatch_slope,1,2)=dischg_for_slope
     $                (imatch_slope,1,2) - GWSMX/86400.
                  ELSE
                     dischg_for_slope(imatch_slope,1,3)=dischg_for_slope
     $                (imatch_slope,1,3) + GWSMX/86400.
                  ENDIF
               ENDIF
            ENDIF
         ENDIF
   20 CONTINUE
      RETURN
      END

C*****************************************************************************
C   This function determines the total surface flow interaction
C   between canal and grid cells through which the canal passes.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION tovflc (n, stagec, ioptn, conv, iterl, ilocal)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'smallres.inc'
      INCLUDE 'dlycslope.inc'

      REAL conv

      DIMENSION ilgth(max_ncnlloc), nodec(max_ncnlloc)
      DOUBLE PRECISION stagec
```

```
      NPTS2 = 0
      VOLINM = 0.0
      DO 10 IN = 1,N-1
         NPTS2 = NPTS2 + NODCR(IN)
   10 CONTINUE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      TOVFLC = 0.0
      DSUM   = 0.0
      DO 80 IC = 1,NODCR(N)
         NPTS2 = NPTS2 + 1
         NODEC(IC) = NODECN(NPTS2) / 10
         ILGTH(IC) = MOD(NODECN(NPTS2) ,  10)
         NODE = NODEC(IC)
         DSUM = DSUM + DISTNC(ILGTH(IC))
         HDIFF = HDC(N,iseas) * (CL(N) - DSUM) / CL(N)
         RCAR = DISTNC(ILGTH(IC)) * WIDTH(N)
         SWLMIN = STAGEC + HDIFF
C-------------------
         IF (ILOCAL .EQ. 1) THEN
            IF (ITERL .EQ. 0) PNDDPTH(N,ILOCAL,IC) = POND(NODE)
         ELSE
            PNDDPTH(N,ILOCAL,IC) = POND(NODE)
         ENDIF
C
         STAGE = PNDDPTH(N,ILOCAL,IC) + ELLS(NODE)
C
         frac_rem_area_cell = 1.0 - total_small_afact(node)
c        if (cnm(n).eq.'L31NC'.and.IOPTN .eq.0.and.ilocal.eq.1
c    $    .and. ITERL .ne. 0) print *,
c    $    PNDDPTH(N,ILOCAL,IC),STAGE,SWLMIN,GDAR/43560.,RCAR/43560.
c    $    ,frac_rem_area_cell,STAGEC,'  INSIDE TOVFLC  '
c    $    ,HDC(N,iseas),CL(N)/5280.,HDIFF,ITERL,ILOCAL,MONTH,IDAY
c    $    ,POND(NODE)
         IF ((LCNB(N).NE.IBSN(NODE).OR. LCNB(N).EQ.88).AND.CNM(N) .NE.
     $   canl_names_spec(35)) GO TO 70
         IF (PNDDPTH(N,ILOCAL,IC) .LE. 0.AND.SWLMIN.LE.ELLS(NODE))
     $   GO TO 70
         IF (STAGE - SWLMIN)  20, 70,30
   20    IF (LUTYP(NODE) .EQ. 0) GO TO 70
         IF (CNM(N) .EQ. canl_names_spec(35) .OR. CNM(N) .EQ.
     $   canl_names_spec(45)) GO TO 70
         IF (IOPTN .EQ.  1) THEN
            IF (conv .EQ. 1.) THEN
               IF (PNDDPTH(N,ILOCAL,IC).LT.-0.010) PRINT *,
     $         'WARNING: PONDING ATCANAL  ',cnm(n),'  NODE = ',NODE,
     $         '  IS LESS THAN ZERO, VALUE =  ',POND(NODE)
            ENDIF
         ENDIF
         HMC=((SWLMIN-ELLS(NODE))*RCAR+PNDDPTH(N,ILOCAL,IC)*(GDAR
     $   *frac_rem_area_cell-RCAR)) / (GDAR*frac_rem_area_cell)
         HMC = AMAX1(HMC,0.0)
         VOLOUT = (SWLMIN - STAGE)*(GDAR*frac_rem_area_cell - RCAR)
         IF (HMC .GT. 0.0) THEN
            dph_c = SWLMIN - STAGE
            ZN = ofm_canl(N,IC,3)* dph_c**ofm_canl(N,IC,4)
```

```
            OFM = 1.49* 86400. / ZN
            VOFC = OFM * SQRT((SWLMIN - STAGE)/(0.5*WIDTH(N))) * HMC **
     $      (2./3.)
            FLVOL = 0.50 * VOFC * DT * DX * HMC
         ELSE
            FLVOL = 0.0
         ENDIF
         VOLINM = - AMIN1(FLVOL,VOLOUT)
         IF (IOPTN .EQ. 1) THEN
            IF (CONV.EQ. 1.) POND(NODE) = POND(NODE) - VOLINM / (GDAR
     $      *frac_rem_area_cell - RCAR)
            IF(CONV.EQ. 1.)THEN
               OVLFLO_TO_CANAL(NODE) = VOLINM/43560.
               rmonthly_ovlflo_to_canal(node) =
     $         rmonthly_ovlflo_to_canal (node) + OVLFLO_TO_CANAL(NODE)
            ENDIF
         ENDIF
         GO TO 70
   30    IF (PNDDPTH(N,ILOCAL,IC) - deten_cnl(n,ic)) 70,70,40
   40    IF (LUTYP(NODE) .EQ. 0) GO TO 70
         IF (SWLMIN .GE. ELLS(NODE)) THEN
            HMC=SWLMIN-ELLS(NODE)
            PDVOL=(PNDDPTH(N,ILOCAL,IC)-AMAX1(HMC,DETEN_CNL(N,IC)))
     $      *(GDAR*frac_rem_area_cell - RCAR)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         ELSE
            PDVOL = (PNDDPTH(N,ILOCAL,IC) - deten_cnl(N,IC)) * (GDAR
     $      * frac_rem_area_cell - RCAR)
         ENDIF
         ZN=ofm_canl(N,IC,1)*PNDDPTH(N,ILOCAL,IC)**ofm_canl(N,IC,2)
         OFM = 1.49* 86400. / ZN
         VOFC= OFM*SQRT((STAGE - SWLMIN)/WIDTH(N))*PNDDPTH(N,ILOCAL,IC)
     $   ** (2./3.)
         FLVOL = 0.50* VOFC * DT * DX * PNDDPTH(N,ILOCAL,IC)
         IF (FLVOL - PDVOL) 50,50,60
   50    VOLINM = FLVOL
         IF (IOPTN .EQ. 1) THEN
            IF (CONV.EQ.1.) POND(NODE) = POND(NODE) - FLVOL / (GDAR
     $      * frac_rem_area_cell - RCAR)
            IF(CONV.EQ.1.)THEN
               OVLFLO_TO_CANAL(NODE) = FLVOL/43560.
               rmonthly_ovlflo_to_canal(node) =
     $         rmonthly_ovlflo_to_canal (node) + OVLFLO_TO_CANAL(NODE)
            ENDIF
         ENDIF
         GOTO 70
   60    IF (IOPTN .EQ. 1) THEN
            IF(CONV.EQ. 1.) POND(NODE) = POND(NODE) - PDVOL/(GDAR
     $         *frac_rem_area_cell - RCAR)
            IF(CONV.EQ. 1.)THEN
               OVLFLO_TO_CANAL(NODE) = PDVOL/43560.
               rmonthly_ovlflo_to_canal(node) =
     $         rmonthly_ovlflo_to_canal (node) + OVLFLO_TO_CANAL(NODE)
            ENDIF
         ENDIF
         VOLINM= PDVOL
   70    TOVFLC = TOVFLC + VOLINM
```

```
         IF (IOPTN .EQ. 1 .and. ivary_slope_daily_opt(n).eq.1) then
            call match(cnm(N),1,5,canal_name_dvslope
     $      ,ncnl_dly_slope,5,imatch_slope)
            IF (IC .LE. mid_point_canl(N)) THEN
               dischg_for_slope(imatch_slope,1,2)=dischg_for_slope
     $          (imatch_slope,1,2) + AMAX1(VOLINM,0.0)/86400.
            ELSE
               dischg_for_slope(imatch_slope,1,3)=dischg_for_slope
     $          (imatch_slope,1,3) - AMAX1(VOLINM,0.0)/86400.
            ENDIF
         ENDIF
         VOLINM = 0.0
   80 CONTINUE
      RETURN
      END

C*****************************************************************************
C   This function converts relative month number to the
C   regular 12-month index.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION monchk (month)

      monchk = month
      IF (monchk .GT. 12) monchk = month - 12

      RETURN
      END

C*****************************************************************************

C   inputs: iday,month     output: sched

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION sched (iday, month)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'opersched.inc'

      INTEGER numd(12)

      REAL regstg1, x(14), y(14)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      xval = julian(month,iday,numd)

      DO 10 j = 1,nbrkpt_ssm
         x(j) = iregjul1(j)
         y(j) = regstg1(j)
   10 CONTINUE

      n = nbrkpt_ssm
      sched = ainterp(xval,x,y,n,0.,0.,1)
      RETURN
      END
```

```
C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C  This function linearly interpolates between break points.
C  Function is typically used to define a calendar-based daily
C  operational line or regulation schedule.
C
      FUNCTION ainterp (xval, x, y, n, xbefore, xafter, itimeopt)

      DIMENSION x(n), y(n)

      IF (xval .LT. x(1) .OR. xval .GT. x(n)) THEN
         IF (itimeopt .EQ. 1) THEN
            i = n
            ip1 = 1
            IF (xval .LT. x(1)) THEN
               xval = x(n) + xafter + xval
            ENDIF
            x1 = x(n) + xafter + xbefore
            ainterp = (xval - x(i))/(x1 - x(i))*(y(ip1) - y(i)) + y(i)

            RETURN
         ELSE
            IF (xval .LT. x(1)) THEN
               i = 1
               GO TO 20
            ELSE IF (xval .GT. x(n)) THEN
               i = n
               ainterp = (xval - x(i))/(x(i) - x(i-1))*(y(i) - y(i-1)) +
     $         y(i)
               RETURN
            ENDIF
         ENDIF
      ENDIF
      DO 10 i = 1,n - 1
         IF (xval .GE. x(i) .AND. xval.LE.x(i+1)) goto 20
   10 CONTINUE
   20 CONTINUE
      ainterp = (xval - x(i))/(x(i+1) - x(i))*(y(i+1) - y(i)) + y(i)

      RETURN
      END

C*****************************************************************************

C   inputs: iday,month,iregjulwt,regstgwt     output: schedwt

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION schedwt (iday, month, nbrkpt_ssmwt, iregjulwt, regstgwt)

      INTEGER numd(12)

      REAL iregjulwt(25), regstgwt(25), x(14), y(14)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/
```

```
      xval = julian(month,iday,numd)

      DO 10 j = 1,nbrkpt_ssmwt
         x(j) = iregjulwt(j)
         y(j) = regstgwt(j)
   10 CONTINUE

      n = nbrkpt_ssmwt
      schedwt = ainterp(xval,x,y,n,0.,0.,1)

      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION schedwn (iday, month, nbrkpt_ssmwn, iregjulwn, regstgwn)

      INTEGER numd(12)

      REAL iregjulwn(25), regstgwn(25), x(14), y(14)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      xval = julian(month,iday,numd)

      DO 10 j = 1,nbrkpt_ssmwn
         x(j) = iregjulwn(j)
         y(j) = regstgwn(j)
   10 CONTINUE

      n = nbrkpt_ssmwn
      schedwn = ainterp(xval,x,y,n,0.,0.,1)

      RETURN
      END
C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION sched1 (iday, month, nbrkpt_ssm1)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'opersched.inc'

      INTEGER numd(12)

      REAL regstg1, x(14), y(14)

      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      xval = julian(month,iday,numd)

      DO 10 j = 1,nbrkpt_ssm1
         x(j) = iregjul2(j)
```

```
         y(j) = regstg2(j)
   10 CONTINUE

      n = nbrkpt_ssm1
      sched1 = ainterp(xval,x,y,n,0.,0.,1)

      RETURN
      END

C*****************************************************************************

C     Given lake okeechobee stage, storage and date this function
C     computes water supply releases to the lake service area. This
C     plan takes effect when the lake stage is projected to fall
C     below 10.5' nvgd during the first two months of the wet season.
C     The function is based on the documentation on the Lake Okeechobee
C     Supply-Side Management Plan (Alan Hall, 1991).
C
C     required input:  1. stglake - lake stage (ft.,ngvd)
C                      2. stolake - lake storage (af)
C                      3. mon     - month
C                      4. iday    - day
C                      5. dmd     - lake service area demand (af/day)
C                      6. dellec  - delivery to lec (af/day)
C                      7. sch     - schedule below which supply-side
C                                   management is imposed
C
C     output:          1. ssmwet - actual delivery or lake service
C                                   area demand met
C
C---  DIMENSION and initialize projected et, rf, and normal water use
C---  for the remaining wet season months of june and july. daymon
C---  defines the number of days in a month.

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      FUNCTION ssmwet (stglake, mon, iday, dmd, dellec, sch,
     & lok_split_option, fract_rain_litzone, wup, lok_targ_level_wet1,
     & lok_targ_level_wet2, ibm_initc, ifreq_init,
     & initialize_annually_opt, iyear, day1, nwgoalsto, jday,
     & mon_targ_wet, iday_targ_wet, ifyr, etp, rfp)

      INCLUDE 'wmm_par.inc'

      REAL daymon(12), dellec, dmd, etp(12), etp_red(12),
     & lok_targ_level_wet1(max_n_goal_ssm),
     & lok_targ_level_wet2(max_n_goal_ssm), rfp(12), rfp_red(12),
     & sch, stglake, wup(12)

      INTEGER iday, iday_targ_wet(max_n_goal_ssm),
     & jdaywgoalsto(max_n_goal_ssm), mon, mon_targ_wet(max_n_goal_ssm),
     & nremnodays(max_n_goal_ssm), numd(12)

      LOGICAL day1, lstartwets

      DATA daymon /31,28,31,30,31,30,31,31,30,31,30,31/
```

```
      IF (mod(ifyr+iyear-1,4).eq.0) THEN
         daymon(2)=29.
      ELSE
         daymon(2)=28.
      ENDIF

      DO i=1,12
         numd(i) = nint(daymon(i))
      ENDDO

      gtotssm = 0.0
      gtotdmd = 0.0

      IF (lok_split_option .EQ. 0) THEN
         stolake = stg2sto(stglake,0)
      ELSE
         stolake = stg2stolitz(stglake,0)
      ENDIF

C---  initialize day counter at the beginning of the wet season (june 1)
C---  and initially set END of 2 months (aug. 1) goal storage to the
C---  water level equivalent to 10.5' ngvd

      lstartwets = (mon .EQ. 6 .AND. iday .EQ. 1)  .OR. ((mon .EQ.
     & ibm_initc .and. mod(iyear-1,ifreq_init) .eq. 0 .and. iday.eq.1
     & .and. initialize_annually_opt .eq. 1) .or. DAY1)
      IF (lstartwets .or. ( mon .ge. 6 .and. mon .le. 9)) then
         IF (lstartwets) THEN
            idaycnt = 0
            jdaywgoalsto(1) = julian(mon_targ_wet(1)
     $      ,iday_targ_wet(1),numd)
            rlok_targ_level1 = lok_targ_level_wet1(1)
            rlok_targ_level2 = lok_targ_level_wet2(1)
            IF (lok_split_option .EQ. 0) THEN
                goalsto = stg2sto(rlok_targ_level1,0)
                goalsto1= stg2sto(rlok_targ_level2,0)
            ELSE
                goalsto = stg2stolitz(rlok_targ_level1,0)
                goalsto1 = stg2stolitz(rlok_targ_level2,0)
            ENDIF
            sumlec   = 0.0
         ELSE
            IF (nwgoalsto .gt. 1) THEN
              DO i = 1,nwgoalsto-1
                jdaywgoalsto(i) = julian(mon_targ_wet(i)
     $           ,iday_targ_wet(i),numd)
                IF (jday .eq. jdaywgoalsto(i)+1) THEN
                   rlok_targ_level1 = lok_targ_level_wet1(i+1)
                   rlok_targ_level2 = lok_targ_level_wet2(i+1)
                   IF (lok_split_option .EQ. 0) THEN
                      goalsto = stg2sto(rlok_targ_level1,0)
                      goalsto1= stg2sto(rlok_targ_level2,0)
                   ELSE
                      goalsto = stg2stolitz(rlok_targ_level1,0)
                      goalsto1 = stg2stolitz(rlok_targ_level2,0)
                   ENDIF
                   sumlec   = 0.0
```

```
                ENDIF
              ENDDO
              jdaywgoalsto(nwgoalsto) = julian(mon_targ_wet
     $        (nwgoalsto),iday_targ_wet(nwgoalsto),numd)
            ENDIF
         ENDIF
      ENDIF
C     print *,lstartwets,rlok_targ_level1,rlok_targ_level2
C    $   ,goalsto,goalsto1,nwgoalsto,mon_targ_wet(1),iday_targ_wet(1)
C    $   ,mon_targ_wet(2),iday_targ_wet(2),jdaywgoalsto(1)
C    $   ,jdaywgoalsto(2),jday
C
      IF (iday .EQ. 1) THEN
            sumlec   = 0.0
         ssmtot=0.0
         dmdtot=0.0
      ENDIF
C
C---  increment wet season day counter
C
      IF (lok_split_option .EQ. 0) THEN
         DO m = 1,12
            rfp_red(m) = rfp(m)
         END DO
      ELSE
         DO m = 1,12
            rfp_red(m) = rfp(m)
            etp_red(m) = etp(m)
            etp(m) = etp_red(m)
         END DO
      ENDIF
      idaycnt = idaycnt + 1
C
C**********************************************************************
C
C     IF beginning of a new week recompute allowable monthly allocations,
C     allowable daily allocations, and actual daily allocations based on
C     allowable daily allocation, available storage, and remaining normal
C     wet-season water use
C
C**********************************************************************
C
      IF ( mod(idaycnt-1,7) .NE. 0) GO TO 10
C
C---  set variables at the beginning of week
C
      monext = monchk(mon+1)
      m2next = monchk(mon+2)
C
C---  compute allowable daily allocation based on allowable
C---     monthly allocation for the current month
C
      IF ( daymon(mon)-iday .GE. 7. ) THEN
C week falls entirely within a single month
         wuday = ( wup(mon)-wup(monext) ) / daymon(mon)
      ELSE
C week INCLUDEs portions of two months, use weighted allocations
```

```
         wuday = ( wup(mon)-wup(monext) ) / daymon(mon) *( daymon(mon)
     $   -(iday-1) ) + ( wup(monext)-wup(m2next) ) / daymon(monext) *(
     $   7.-(daymon(mon)-(iday-1)) )

         wuday = wuday/7.0
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C---  calculate available storage based on storage above goal storage,
C---  previous week 7-day delivery to lec, and anticipated et & rf for
C---  the remainder of the 2-month wet season
C
C lower goal by sum of last week's discharges made to the lec
C
      goalsto = goalsto - sumlec
      goalsto1 = goalsto1 - sumlec
C     PRINT *,sumlec
C
C reset sum of lec delivery for the week to zero
C
      sumlec = 0.0
C
C compute available storage above 2-month wet season END goal storage
C
      avail = stolake - goalsto
      avail1 = stolake - goalsto1
C     print *,avail,stolake,goalsto,wuday,wup(mon),wup(monext),' PPPP '
C    $  ,rfp(mon),etp(mon),rfp(monext),etp(monext)
C
C adjust the available storage to account for the net effect of et and rf
C
      etmon = etp(mon) - etp(monext)
      rfmon = rfp_red(mon) - rfp_red(monext)
C
      adjust = (etp(mon)-rfp_red(mon)) - (etmon-rfmon)/daymon(mon)
     $*(iday-1)
      avail = avail - adjust
      avail1 = avail1 - adjust
      IF ( avail .LT. 0.0)  avail  = 0.0
      IF ( avail1 .LT. 0.0)  avail1 = 0.0
C     remain = wup(mon) - (wup(mon)-wup(monext))/daymon(mon)*(iday-1)
C
C---  calculate normal use for the remainder of the 2-month wet season
C
      remuse = (wup(mon) - (wup(mon)-wup(monext))/daymon(mon)*(iday-1))
C
C---  compute actual daily allocation (supply) for this week
C---  calculate supply based on wuday, avail, and remuse
C
      supply = wuday * avail/remuse
      supply1 = wuday * avail1/remuse
C     ratio = avail/remuse
C      PRINT *,supply,supply1,wuday,avail,avail1,remuse,mon,iday,' LKLK'
      IF ( supply .GT. wuday ) supply = wuday
      IF ( supply1 .GT. wuday ) supply1 = wuday
      IF (nwgoalsto .gt. 1) THEN
         DO i = 1,nwgoalsto
```

```
            nremnodays(i) = jdaywgoalsto(i)-jday
            IF (nremnodays(i) .ge. 0 .and. nremnodays(i) .lt. 7
     $      .and. i .ne. nwgoalsto) then
               supply = supply * 7.0/nremnodays(i)
               supply1 = supply1 * 7.0/nremnodays(i)
            ENDIF
         ENDDO
      ENDIF
C
   10 CONTINUE
C
C---  sum discharges made to lower east coast everyday of this week
C
      sumlec = sumlec + dellec
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C***********************************************************************
C
C     IF lake stage is greater than ssm schedule RETURN
C
C***********************************************************************
C
      IF ( stglake .GT. sch )THEN
         ssmwet = dmd
C        dmet = 1.0
C
         ssmtot = ssmtot + ssmwet
         dmdtot = dmdtot + dmd
C        IF (dmdtot .GT. 0.0) THEN
C           dmettot = ssmtot/dmdtot
C        ELSE
C           dmettot = 1.0
C        ENDIF
C         PRINT *,mon,iday,supply,ssmwet,dmd,stglake,sch
         IF (iday .EQ. daymon(mon)) THEN
            gtotssm = gtotssm + ssmtot
            gtotdmd = gtotdmd + dmdtot
C            PRINT *,'MONTH TOTALS: ',ssmtot,dmdtot,dmettot
         ENDIF
         RETURN
C         goto 201
      ENDIF
C
C***********************************************************************
C
C supply  - actual normal daily allocation for this week
C
C***********************************************************************
C next 5 lines for info only
      IF ( supply .GT. dmd )THEN
         surplus = supply - dmd
      ELSE
         short = dmd - supply
      ENDIF
C
C***********************************************************************
C
```

```
C     compute actual daily delivery (i.e. today's water supply release)
C
C     ssmwet is the actual daily delivery or demand met using supply-
C     side management. it is the minimum between actual daily allocation
C     and the lake service area demand for the day.
C
C*********************************************************************
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      ssmwet = amin1(supply, dmd)
      IF (dmd .GT. 0.0) THEN
         wrspdmd = ssmwet/dmd
      ELSE
         wrspdmd = 1.0
      ENDIF
C
      IF (wrspdmd .LT. 0.33) THEN
C use ssmwet based on lower target IF cutback is more than 67%
         ssmwet = amin1( supply1, dmd)
      ENDIF
C      PRINT *,wrspdmd,supply1,dmd,mon,iday,ssmwet
      ssmtot = ssmtot + ssmwet
      dmdtot = dmdtot + dmd
C     IF (dmdtot .GT. 0.0) THEN
C        dmettot = ssmtot/dmdtot
C     ELSE
C        dmettot = 1.0
C     ENDIF
C
C     IF (dmd .GT. 0.0) THEN
C        dmet = ssmwet/dmd
C     ELSE
C        dmet = 1.0
C     ENDIF
C
      IF (iday.EQ.daymon(mon)) THEN
         gtotssm = gtotssm + ssmtot
         gtotdmd = gtotdmd + dmdtot
C         PRINT *,'MONTH TOTALS: ',ssmtot,dmdtot,dmettot
C        ssmtot = 0.0
C        dmdtot = 0.0
      ENDIF
      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION lvseepcfs (nodelsp)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
```

```
      INCLUDE 'lakwca.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'

      LL = 0
      iexist_levee = 0
      DO lv = 1,nlevs
         DO n = 1,lvnodes(lv)
            LL = LL + 1
            LNODE1 = LVSP(LL,1) - MINX(LVSP(LL,2)) +  1+ ISUM
     $      (LVSP(LL,2))
            IF (LNODE1 .EQ. NODELSP) THEN
               L = LL
               IV = LV
               iexist_levee = 1
            ENDIF
         END DO
      END DO
      IF (iexist_levee .EQ. 0) THEN
         LVSEEPCFS = 0.0
      ELSE
         N3 = LVSP(L,1) - MINX(LVSP(L,2)) +  1+ ISUM(LVSP(L,2))
         N4 = LVSP(L,6) - MINX(LVSP(L,7)) +  1+ ISUM(LVSP(L,7))
         N4_EAST = N4 + 1
         GO TO (10,20,30,40) LVSP(L,3)
   10    N1 = LVSP(L,1) - MINX(LVSP(L,2)) +  1+ ISUM(LVSP(L,2))
         N2 = LVSP(L,5)
CBW
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         STGDIFF_CANAL_DN = FLOAT(NODCR(N2)-ICNL_NODE_POS(L,2))/ FLOAT
     $   (NODCR(N2)) * HDC(N2,iseas)
         CANAL_STAGE_DN = CHDEP(N2) + STGDIFF_CANAL_DN
         IF (LVSP(L,1) .EQ. 26 .AND. LVSP(L,2) .EQ.  22.AND. CNM(N2)
     $   .EQ. canl_names_spec(27) .AND. IBSN(N4) .EQ. 13.AND. IBSN
     $   (N4_EAST) .EQ. 0) THEN
            STGDN = CHDEP(IC4) + HDC(IC4,iseas)
         ELSEIF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 27) THEN
            STGDN = CHDEP(ISNCRE) + HDC(ISNCRE,iseas)
         ELSE
            STGDN = CANAL_STAGE_DN
         ENDIF
C
         DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
         IF (POND(N1) .GT. DPTHGW) THEN
            HW_SEEP = POND(N1) - DPTHGW + ELLS(N1)
         ELSE
            HW_SEEP = (POND(N1)-DPTHGW)/S(N1) + ELLS(N1)
         ENDIF
         DPTHGW = (ELLS(N4) - H(N4)) * S(N4)
         IF (POND(N4) .GT. DPTHGW) THEN
            HDIFF3 = POND(N4) - DPTHGW + ELLS(N4)
         ELSE
            HDIFF3 = (POND(N4)-DPTHGW)/S(N4) + ELLS(N4)
         ENDIF
         HDIFF = HW_SEEP - STGDN
         HDIFF1 = HW_SEEP
```

```
         HDIFFU = HDIFF1 - HDIFF3
         GO TO 50
   20    N1 = LVSP(L,4)
         N2 = LVSP(L,6) - MINX(LVSP(L,7)) +  1+ ISUM(LVSP(L,7))
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         STGDIFF_CANAL_UP = FLOAT(NODCR(N1)-ICNL_NODE_POS(L,1))
     $   / FLOAT(NODCR(N1)) * HDC(N1,iseas)
         CANAL_STAGE_UP = CHDEP(N1) + STGDIFF_CANAL_UP
C
         DPTHGW = (ELLS(N3) - H(N3)) * S(N3)
         IF (POND(N3) .GT. DPTHGW) THEN
            HDIFF1 = POND(N3) - DPTHGW + ELLS(N3)
         ELSE
            HDIFF1 = (POND(N3)-DPTHGW)/S(N3) + ELLS(N3)
         ENDIF
         DPTHGW = (ELLS(N2) - H(N2)) * S(N2)
         IF (POND(N2) .GT. DPTHGW) THEN
            HDIFF3 = POND(N2) - DPTHGW + ELLS(N2)
         ELSE
            HDIFF3 = (POND(N2)-DPTHGW)/S(N2) + ELLS(N2)
         ENDIF
         HDIFF = CANAL_STAGE_UP - HDIFF3
         HDIFFU = HDIFF1 - HDIFF3
         GO TO 50
   30    N1 = LVSP(L,4)
         N2 = LVSP(L,5)
         STGDIFF_CANAL_DN = FLOAT(NODCR(N2)-ICNL_NODE_POS(L,2))/
     $   FLOAT(NODCR(N2)) * HDC(N2,iseas)
         CANAL_STAGE_DN = CHDEP(N2) + STGDIFF_CANAL_DN
         STGDIFF_CANAL_UP = FLOAT(NODCR(N1)-ICNL_NODE_POS(L,1))
     $   / FLOAT(NODCR(N1))* HDC(N1,iseas)
         CANAL_STAGE_UP = CHDEP(N1) + STGDIFF_CANAL_UP
         HDIFF = CANAL_STAGE_UP - CANAL_STAGE_DN
         HDIFF_MAX = HDIFF
         GO TO 50
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
   40    N1 = LVSP(L,1) - MINX(LVSP(L,2)) +  1 + ISUM(LVSP(L,2))
         N2 = LVSP(L,6) - MINX(LVSP(L,7)) +  1 + ISUM(LVSP(L,7))
         DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
         IF (POND(N1) .GT. DPTHGW) THEN
            STGUP = POND(N1) - DPTHGW + ELLS(N1)
         ELSE
            STGUP = (POND(N1)-DPTHGW)/S(N1) + ELLS(N1)
         ENDIF
         DPTHGW = (ELLS(N2) - H(N2)) * S(N2)
         IF (POND(N2) .GT. DPTHGW) THEN
            STGDN = POND(N2) - DPTHGW + ELLS(N2)
         ELSE
            STGDN = (POND(N2)-DPTHGW)/S(N2) + ELLS(N2)
         ENDIF
         HDIFF = STGUP - STGDN
         HDIFFU = HDIFF
   50    IF (SRATE(IV,1) .LT. 0.0) THEN
            HD_DROP = AMAX1(HDIFF,0.0)
         ELSE
            HD_DROP = HDIFF
         ENDIF
```

```
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         IF (SRATE(IV,2) .LT. 0.0) THEN
            HD_DROP_N = AMAX1(HDIFFU,0.0)
         ELSE
            HD_DROP_N = HDIFFU
         ENDIF
         SEEPCFS = SRATE(IV,1)*HD_DROP+SRATE(IV,2)*HD_DROP_N + SRATE
     $   (IV,3)
         IF(LVNAME(IV) .EQ. 'L40E ') THEN
            IF (SEEPCFS .LT. -0.66) SEEPCFS = -0.66
         ELSEIF (LVNAME(IV) .NE. 'L38E ') THEN
            IF (SEEPCFS .LT. 0.0) SEEPCFS = 0.0
         ENDIF
         SEEPCFS = SEEPCFS * srate_frac(IV)
         SEEPCFS = amin1(SEEPCFS,rate_limit(IV))
         LVSEEPCFS = SEEPCFS * DISTNC(LVSP(L,8))/5280.
      ENDIF
      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2sto (stgto, iflag)

      REAL stage(54), storag(54)

      DATA stage /0.0,1.0,2.0,3.0,4.0,5.0,6.0,7.0, 8,8.5,9,9.5,10,10.5,
     $   11,11.5,12,12.5,13, 13.5,14,14.5,15,15.5,16,16.5,17,17.5, 18,
     $   18.5,19,19.5,20,20.5,21,21.5,22,22.5, 23,23.5,24,24.5,25,25.5,
     $   26,26.5,27,27.5, 28,28.5,29,29.5,30,50/
      DATA storag /0,50000,150000,300000,420000,680000,920000,
     $   1170000, 1442000, 1586000, 1729000, 1884000, 2039000, 2203000,
     $   2366000, 2544000, 2722000, 2915000, 3108000, 3317000, 3527000,
     $   3755000, 3980000, 4200000, 4425000, 4650000, 4875000, 5105000,
     $   5335000, 5565000, 5790000, 6030000, 6260000, 6495000, 6730000,
     $   6965000, 7195000, 7435000, 7670000, 7920000, 8150000, 8400000,
     $   8640000, 8890000, 9140000, 9390000, 9640000, 9890000, 10140000,
     $   10400000, 10650000, 10910000, 11160000, 20240000/

      IF (IFLAG .EQ. 0) THEN
         STG2STO = AINTERP(STGTO,STAGE,STORAG,54,0.,0.,0)
      ELSE
         STG2STO = AINTERP(STGTO,STORAG,STAGE,54,0.,0.,0)
      ENDIF

      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2stolitz (stgto, iflag)
```

```
      REAL stage(54), storag(54)

      DATA stage /0.0,1.0,2.0,3.0,4.0,5.0,6.0,7.0, 8,8.5,9,9.5,10,10.5,
     $   11,11.5,12,12.5,13, 13.5,14,14.5,15,15.5,16,16.5,17,17.5, 18,
     $   18.5,19,19.5,20,20.5,21,21.5,22,22.5, 23,23.5,24,24.5,25,25.5,
     $   26,26.5,27,27.5, 28,28.5,29,29.5,30,50/
      DATA storag /0,7.0,20.0,3814.0,21943.0,93219.0,169492.0,
     $   254070.0, 355420.0,412838.0,471908.0,538088.0,606537.0,
     $   681466.0,758676.0,846479.0,937808.0,1039993, 1144671, 1261538,
     $   1382892, 1518095, 1653095, 1785095, 1920095, 2055095, 2190095,
     $   2328095, 2466095, 2604095, 2739095, 2883095, 3021095, 3162095,
     $   3303095, 3444095, 3582095, 3726095, 3867095, 4017095, 4155095,
     $   4305095, 4449095, 4599095, 4749095, 4899095, 5049095, 5199095,
     $   5349095, 5505095, 5655095, 5811095, 5961095, 11409095/

      IF (IFLAG .EQ. 0) THEN
         STG2STOLITZ = AINTERP(STGTO,STAGE,STORAG,54,0.,0.,0)
      ELSE
         STG2STOLITZ = AINTERP(STGTO,STORAG,STAGE,54,0.,0.,0)
      ENDIF

      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION stg2stores (stgto, iflag)

      REAL stage(54), storag(54)

      DATA stage /0.0,1.0,2.0,3.0,4.0,5.0,6.0,7.0, 8,8.5,9,9.5,10,10.5,
     $   11,11.5,12,12.5,13, 13.5,14,14.5,15,15.5,16,16.5,17,17.5, 18,
     $   18.5,19,19.5,20,20.5,21,21.5,22,22.5, 23,23.5,24,24.5,25,25.5,
     $   26,26.5,27,27.5, 28,28.5,29,29.5,30,50/
      DATA storag /0,49993,149980,296186,398057,586781,750508, 915930,
     $   1086580, 1173162, 1257092, 1345912, 1432463, 1521534, 1607324,
     $   1697521, 1784192, 1875007, 1963329, 2055462, 2144108, 2236905,
     $   2326905, 2414905, 2504905, 2594905, 2684905, 2776905, 2868905,
     $   2960905, 3050905, 3146905, 3238905, 3332905, 3426905, 3520905,
     $   3612905, 3708905, 3802905, 3902905, 3994905, 4094905, 4190905,
     $   4290905, 4390905, 4490905, 4590905, 4690905, 4790905, 4894905,
     $   4994905, 5098905, 5198905, 8830905/

      IF (IFLAG .EQ. 0) THEN
         STG2STORES = AINTERP(STGTO,STAGE,STORAG,54,0.,0.,0)
      ELSE
         STG2STORES = AINTERP(STGTO,STORAG,STAGE,54,0.,0.,0)
      ENDIF

      RETURN
      END

C*****************************************************************************

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
      INTEGER FUNCTION iag (ibsn)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'agdata.inc'

      DO 10 INDEX = 1,NAGPTS
         IF( IBSN .EQ. IAGFLO(INDEX,2)) THEN
            IAG = INDEX
            RETURN
         ENDIF
   10 CONTINUE

      PRINT *,ibsn,IAGFLO(INDEX,2)
      PRINT *,' ILLEGAL EVERGLADES AGRICULTURAL AREA INDEX'
      PRINT *,' ABORT IN FUNCTION IAG-- CALLED BY ROUTE'
      STOP 12

      END

C*****************************************************************************

C   This function computes the number of months between two dates.

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      INTEGER FUNCTION nmosdiff (imo1, iyr1, imo2, iyr2)

      INTEGER iyr(2), mon(2)

      mon(1) = imo1
      iyr(1) = iyr1

      mon(2) = imo2
      iyr(2) = iyr2

      nmosdiff = (12-(mon(1)-1)) + mon(2) + ((iyr(2)-iyr(1)-1)*12) -1

      RETURN
      END

C*****************************************************************************

C function est_etiu RETURNs an estimate of monthly unsaturated zone et
C for 1 of 6 irrigated use types (urban landscape, nursery, golf course,
C ag-lowvolume, ag-overhead, and ag-other).
C Input and Output Units = inches per month.
C Regression equation coeffiecients developed by Dick March in Dec 1993.
C The equations mimick the AFSIRS model and are used primarily to compute
C the reduction in unsaturated zone ET resulting from a water shortage.
C
Cbw same prefix can only appear once in a statement NTFlint

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      REAL FUNCTION est_etiu (itype, net_irrig, mon_rf, mon_pet)
```

```
      REAL mon_pet, mon_rf, net_irrig

      DIMENSION a(6), b(6), c(6), d(6)

      DATA  a /0.8742, 1.3482, 1.2929, 1.0715, 1.2114, 1.4229/
      DATA  b /0.1951, 0.5752, 0.5129, 0.1318, 0.4976, 0.6475/
      DATA  c /0.1506, 0.1898, 0.1285, 0.1920, 0.2705, 0.1685/
      DATA  d /0.7908, 0.3269, 0.9855, 0.6679, 0.8013, 0.2723/

C     This function computes  evapotranspiration from the unsaturated
C     zone, given: 1) the irrigation type, 2) net monthly irrigation,
C     3)monthly rainfall, and 4)monthly potential evapotranspiration.
C     The  equation takes the form:

C     est_etiu = a  * net_irrig ** b * mon_rf ** c * mon_pet ** d

C     where  a,b,c,d are curve fitting parameters that are a function
C     of  irrigation type (type)   and   est_etiu  is  the  estimated
C     irrigation use  or evapotranspiration from the unsaturated zone.

C     itype = 1 indicates urban landscape
C     itype = 2 indicates nursery
C     itype = 3 indicates golf course
C     itype = 4 indicates agricultural low volume
C     itype = 5 indicates agricultural overhead
C     itype = 6 indicates other agricultural

      IF ( net_irrig .EQ. 0.0 ) net_irrig = .01
      IF ( mon_rf    .EQ. 0.0 ) mon_rf    = .01
      IF ( mon_pet   .EQ. 0.0 ) mon_pet   = .01

      est_etiu = a(itype) * net_irrig**b(itype) * mon_rf**c(itype) *
     & mon_pet**d(itype)

      RETURN
      END
```

gen_canl_dep_struc_flw.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: gen_canl_dep_struc_flw.F,v 1.20 2005/06/10 13:29:06 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/gen_canl_dep_struc_flw.F,v $
C   $Name: rel-5-5 $
C
C    THIS SUBROUTINE CALCULATES FLOW THROUGH OUTLET STRUCTURES
C    FROM A CANAL USING GENERIC CODE
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE gen_canl_dep_struc_flw (n, ind1, idncl, crmin, rain,
     & tivolwtr, capacws, capacfl, prevqu, volwtr, tdqu_dn,
     & outflow_from_3a, outflow_from_3a_temp, temp_volwtr, tempf_volwtr,
     & temp_weircff, tempf_weircff, tempvol, total_small_afact,
     & rloc_excess_for_ws, s334_fc, rloc_excss_to_canl_next_step,
     & node_wd, logdisg, is334_open_flg, conv, jday, iter,
     & tot_avl_l8_bpit_res_lwd1, flow_to_ca1_frm_acmebsn, tdmnd,
     & finstor, finstor_res, flow_to_c9r_frm_c9, flow_to_lbn_frm_c11,
     & wratio, tot_flw_to_c9r_frm_nlkblt, avail_fc_vol,
     & flow_to_cpbres_frm_e1, dischg_to_bay_s29_frm_c11r,
     & frac_avail_cap_s334, cout, wsdmnd, imgt, hws, tws,
     & gravity_flow_struc, flow_to_c9r_frm_c11, tot_grav_cap,
     & s336_to_biscbay, flow_to_acmres_frm_basin, stgdiff_canal_tw,
     & qinfldn, coutlim, avvol_dns334, add_s333_fc_s334,
     & tot_avl_l8_bpit_res_m1q,s337fc,outf_s9a)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
```

```
      INCLUDE 'stas.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'et.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'lokres.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'dualops.inc'

      DIMENSION capacfl(max_n_seas,max_n_substr), capacws(max_n_seas),
     & crmin(max_ncnls,max_n_seas), etout(max_ncnls),
     & ilgth(max_ncnlloc), node_wd(max_ncnls,max_n_weir_outf),
     & nodec(max_ncnlloc), outflow_from_3a(max_n_conv_cnl_wca),
     & outflow_from_3a_temp(max_n_conv_cnl_wca), prevqu(max_ncnls),
     & rain(max_ncells), rfin(max_ncnls),
     & rloc_excess_for_ws(max_n_serv_area),
     & rloc_excss_to_canl_next_step(max_ncnls), tdqu_dn(max_ncnls),
     & temp_volwtr(max_n_res), temp_weircff(max_n_res,max_nostr_res),
     & tempf_volwtr(max_n_res), tempf_weircff(max_n_res,max_nostr_res),
     & tivolwtr(max_n_res), total_small_afact(max_ncells),
     & volwtr(max_n_res), wratio(max_ncnls)

      LOGICAL logdisg

      REAL mean_land_surf_along_canal

      DOUBLE PRECISION crmin,swl_land_surf,swl,hws,tws,hw,tw

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      iflo=istr(n,ind1)
      IF (flnm(istr(n,ind1)).eq.strname_spec(67)) THEN
        s331_pump=0.0
      END IF
      hw=hws
      tw=tws

      DO 20 ns=1,nsubstr(n,ind1)
        IF (ipurpse(n,ind1,ns).ne.1.and.ispec(n,ind1).eq.0) THEN
          IF (logdisg) THEN

            IF (irating(n,ind1,ns,imgt).eq.1) THEN
              IF (stg_trig_fc_ops(n,ind1).gt.rmxstg(n,ind1,iseas)) THEN
                coutfl=capacfl(imgt,ns)
              ELSE IF (stg_trig_fc_ops(n,ind1).gt.
     &        rmnstgs(n,ind1,iseas)) THEN
                pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**
     &           3*(stg_trig_fc_ops(n,ind1)-rmnstgs(n,ind1,iseas))**3
                coutfl=pctgo*capacfl(imgt,ns)
              ELSE
                coutfl=0.0
              END IF
```

```
            ELSE
              coutfl=capacfl(imgt,ns)
            END IF
            IF (opt_fc_canal(n).eq.'FLWTH') coutfl=amin1(coutfl,
     &       avail_fc_vol)
C
          ELSE
            coutfl=0.0
          END IF
c         if (conv.eq.1.) then
c         if (cnm(n).eq.'C51 ') write(445,*)
c    &     stg_trig_fc_ops(n,ind1),rmxstg(n,ind1,iseas)
c    &     ,rmnstgs(n,ind1,iseas),coutfl,hw,tw,flnm(iflo)
c    &     ,capacfl(imgt,ns),month,iday,ifyr+iyear-1,LGRAV(N,IND1,NS)
c    &     ,'  C51OUTF'
c         endif
          if (flnm(iflo).eq.'S335 ' .and. nint(s337fc) .gt. 0)
     &     coutfl = min(coutfl,s337fc)
c           if (conv.eq.1. .and. flnm(iflo).eq.'S380R') print *,HW,
c    $       TW,RMXSTG(N,IND1,ISEAS),CAPACFL(IMGT,NS),COUTFL,
c    $       '  S380FL ',CAPAC,CAPACFLDC,LGRAV(N,IND1,NS),
c    $       RMNSTGS(N,IND1,ISEAS),cnm(n),HDC(N,iseas),
c    $       stg_trig_fc_ops(n,ind1)
          IF (flnm(iflo).eq.strname_spec(16)) THEN
            ws_to_c6w_for_c2c4=amax1(ws_to_c6w-dqu(idncl),0.0)
            coutfl=coutfl-amin1(ws_to_c6w_for_c2c4,coutfl)
          END IF
          IF (flnm(iflo).eq.strname_spec(17)) THEN
            IF (nint(ws_to_c6w).gt.0) THEN
              coutfl=0.0
            END IF
          END IF
          IF (flnm(iflo).eq.strname_spec(18).or.flnm(iflo)
     &     .eq.strname_spec(19).or.flnm(iflo).eq.strname_spec(20)) THEN
            CALL match (canl_names_spec(19), 1, 5, cnm, nch+1, 5, ic6)
            IF (nint(dqu(ic6)).gt.0) THEN
              coutfl=0.0
            END IF
          END IF
          IF (flnm(iflo).eq.'S335  ') THEN
            IF (nint(dqu(idncl))
     &       .eq.0.and.iopt_for_reg_releases_s334.ne.0) THEN
              DO iss=1,nstrctr(idncl)
                IF (flnm(istr(idncl,iss)).eq.'S356IO') THEN
                  IF (irceive(idncl,iss).ne.1) THEN
                    IF (stage3b.ge.8.5) THEN
                      coutfl=amin1(kflo(ifls337),capacfl(imgt,ns))
                    END IF
                  END IF
                END IF
              END DO
            END IF
          END IF
          IF (irating(n,ind1,ns,imgt).ne.2) THEN
            coutlim=0.0
            IF (idncl.ne.0.and.idncl.le.nch) THEN
              DO nd=1,nout(idncl)
```

```
            IF (into(idncl,nd).ge.0) THEN
              IF (into(idncl,nd).ne.999) THEN
                IF (hw.gt.crel(idncl,nd)+stgdiff_canal_tw) THEN
                  IF (into(idncl,nd).eq.0) THEN
                    coutdn=strcf(idncl,nd)*gwdth(idncl,nd)*(hw-
&                    crel(idncl,nd)-stgdiff_canal_tw)*sqrt(hw-
&                    crel(idncl,nd)-stgdiff_canal_tw)
                  ELSE
                    coutdn=strcf(idncl,nd)*gwdth(idncl,nd)*sqrt(hw-
&                    crel(idncl,nd)-stgdiff_canal_tw)
                  END IF
                ELSE
                  coutdn=0.0
                END IF
                coutlim=coutlim+coutdn
              ELSE
                crelev=crel(idncl,nd)
                lstnode=node_wd(idncl,nd)
                deptgw=ells(lstnode)-h(lstnode)
                IF (pond(lstnode).gt.deptgw) THEN
                  tw=pond(lstnode)-deptgw+ells(lstnode)
                ELSE
                  tw=(pond(lstnode)-deptgw)/s(lstnode)+ells(lstnode)
                END IF
                IF (tw.le.crelev) THEN
                  IF (hw.gt.crelev) THEN
                    vol_lim_w=(hw-ells(lstnode))*gdar*0.80
                    coutdn=strcf(n,nd)*86400.*gwdth(n,nd)*(hw-
&                    crelev)*sqrt(hw-crelev)
                    coutdn=amin1(coutdn,vol_lim_w)
                  ELSE
                    coutdn=0.0
                  END IF
                ELSE
                  IF (hw.lt.crelev) THEN
                    sgn=-1.0
                    vol_lim_w=amax1(pond(lstnode)-deten(lstnode),
&                    0.0)*gdar
                    coutdn=sgn*strcf(n,nd)*86400.*gwdth(n,nd)*(tw-
&                    crelev)*sqrt(tw-crelev)
                    coutdn=sgn*amin1(sgn*coutdn,vol_lim_w)
                  ELSE
                    IF (hw.gt.tw) THEN
                      submerg_fact=(tw-crelev)/(hw-crelev)
                      rel_crst_length=(hw-crelev)/(hw-crelev+5.0)
                      rmod_limit=0.85-0.50*rel_crst_length
                      reduc_fact=(submerg_fact-rmod_limit)/(1.0-
&                      rmod_limit)
                      reduc_fact=amax1(reduc_fact,0.0)
                      reduc_fact=(1.0-reduc_fact)**0.25
                      vol_lim_w=(hw-tw)*gdar*0.80
                      coutdn=strcf(n,nd)*86400.*gwdth(n,nd)*(tw-
&                      crelev)*sqrt(tw-crelev)*reduc_fact
                      coutdn=amin1(coutdn,vol_lim_w)
                    ELSE
                      sgn=-1.0
                      submerg_fact=(hw-crelev)/(tw-crelev)
```

```
                        rel_crst_length=(tw-crelev)/(tw-crelev+5.0)
                        rmod_limit=0.85-0.50*rel_crst_length
                        reduc_fact=(submerg_fact-rmod_limit)/(1.0-
     &                   rmod_limit)
                        reduc_fact=amax1(reduc_fact,0.0)
                        reduc_fact=(1.0-reduc_fact)**0.25
                        coutdn=sgn*strcf(n,nd)*86400.*gwdth(n,nd)*(tw-
     &                   crelev)*sqrt(tw-crelev)*reduc_fact
                        vol_lim_w=amax1(pond(lstnode)-deten(lstnode),
     &                   0.0)*gdar*0.9
                        coutdn=sgn*amin1(sgn*coutdn,vol_lim_w)
                      END IF
                    END IF
                  END IF
                  coutlim=coutlim+coutdn
                END IF
              END IF
            END DO
C
            tot_pump=0.0
            IF (idncl.lt.n) THEN
              qinfldn=max(prevqu(idncl)-prev_outf_to_ds_cnl(n,idncl)
     &        *86400.,0.0)
c             if (flnm(iflo).eq.'S381 ')
c    &        print *,qinfldn/86400.,prevqu(idncl)/86400.
c    &        ,prev_outf_to_ds_cnl(n,idncl),cnm(idncl)
            ELSE
              qinfldn=qu(idncl)
            END IF
            DO nd=1,nstrctr(idncl)
              coutlim1=strout(hw,n,idncl,nd,0,crmin,volwtr,
     &         prevqu,rain,iter,tot_pump,jday,tempf_volwtr,
     &         tempf_weircff,0)
              coutlim=coutlim+coutlim1
c             if (flnm(iflo).eq.'S381 ') print *,hw,coutlim
c    &         ,coutlim1,cnm(idncl),nd,'  COUTLIM'
            END DO
C
            IF ((lcnb(idncl).eq.0.and.cnm(idncl)
     &       .ne.canl_names_spec(42)).or.lcnb(idncl)
     &       .eq.11.or.lcnb(idncl).eq.88.or.cnm(idncl)
     &       .eq.canl_names_spec(30).or.cnm(idncl)
     &       .eq.canl_names_spec(29).or.cnm(idncl)
     &       .eq.canl_names_spec(31).or.cnm(idncl)
     &       .eq.canl_names_spec(27).or.cnm(idncl)
     &       .eq.canl_names_spec(32).or.cnm(idncl).eq.'L67E ') THEN
              if (nodcr(idncl) .gt. 1) then
                str_ds_stg_diff_frm_end = float(nodcr(idncl)
     &          - nodepos_sd(n,ind1))/float(nodcr(idncl)-1)
     &          *hdc(idncl,iseas)
              else
                str_ds_stg_diff_frm_end = 0.0
              endif
              avvoldn=(hw-str_ds_stg_diff_frm_end-chdep(idncl))*
     &         cl(idncl)*width(idncl)-cgseep(idncl,hw-
     &         str_ds_stg_diff_frm_end,0,0.)-tovflc(idncl,hw-
     &         str_ds_stg_diff_frm_end,0,0.,iter,2)-qinfldn+coutlim
```

```
     &           *86400.
c                if (flnm(iflo).eq.'S381 ' .and. conv.eq.1.) print *,
c    &           amax1(avvoldn,0.0)/86400.,hw,qinfldn/86400.,coutlim,
c    &           ' S381DN ',chdep(idncl),str_ds_stg_diff_frm_end,
c    &           cgseep(idncl,hw-str_ds_stg_diff_frm_end,0,0.)/86400.,
c    &           tovflc(idncl,hw-str_ds_stg_diff_frm_end,0,0.,iter,2)
c    &           /86400.,cnm(idncl)
                avvoldn=amax1(avvoldn,0.0)/86400.
              ELSE
                npts2=0
                cstor=0.0
                DO in=1,idncl-1
                  npts2=npts2+nodcr(in)
                END DO
C
                rfin(idncl)=0.0
                etout(idncl)=0.0
                dsum=0.0
                voldn=0.0
                mean_land_surf_along_canal=0.0
                DO ic=1,nodcr(idncl)
                  npts2=npts2+1
                  nodec(ic)=nodecn(npts2)/10
                  ilgth(ic)=mod(nodecn(npts2),10)
                  etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/
     &             12.0
                  rf=rain(nodec(ic))/(12.)
                  rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*
     &             width(idncl)
                  etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &             width(idncl)
                  dsum=dsum+distnc(ilgth(ic))
                  hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
                  rcar=distnc(ilgth(ic))*width(idncl)
                  mean_land_surf_along_canal=mean_land_surf_along_canal+
     &             ells(nodec(ic))/nodcr(idncl)
                END DO
                swl_land_surf=mean_land_surf_along_canal-hdc(idncl,
     &           iseas)/2.0
C
                coutlim_ls=0.0
                IF (into(idncl,1).ge.0) THEN
                  DO nd=1,nout(idncl)
                    IF (swl_land_surf.gt.crel(idncl,nd)) THEN
                      IF (into(idncl,1).eq.0) THEN
                        coutdn_ls=strcf(idncl,nd)*gwdth(idncl,nd)*
     &                   (swl_land_surf-crel(idncl,nd))*
     &                   sqrt(swl_land_surf-crel(idncl,nd))
                      ELSE
                        coutdn_ls=strcf(idncl,nd)*gwdth(idncl,nd)*
     &                   sqrt(swl_land_surf-crel(idncl,nd))
                      END IF
                    ELSE
                      coutdn_ls=0.0
                    END IF
                    coutlim_ls=coutlim_ls+coutdn_ls
                  END DO
```

```
                END IF
                tot_pump=0.0
                DO nd=1,nstrctr(idncl)
                  coutlim_ls1=strout(swl_land_surf,n,idncl,nd,0,crmin,
     &             volwtr,prevqu,rain,iter,tot_pump,jday,tempf_volwtr,
     &             tempf_weircff,0)
                  coutlim_ls=coutlim_ls+coutlim_ls1
                END DO
                avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
     &           width(idncl)
                avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)
     &           -abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-qinfldn+
     &           coutlim_ls*86400.-rfin(idncl)+etout(idncl)
                avail_cstor=amax1(avail_cstor,0.0)/86400.
c               swl=chdep(n)
                swl=hw
                if (nodcr(idncl) .gt. 1) then
                  str_ds_stg_diff_frm_end = float(nodcr(idncl)
     &            - nodepos_sd(n,ind1))/float(nodcr(idncl)-1)
     &            *hdc(idncl,iseas)
                else
                  str_ds_stg_diff_frm_end = 0.0
                endif
                IF (swl.gt.swl_land_surf) THEN
                  IF (tovflc(idncl,swl,0,0.,iter,2).lt.0.0) THEN
                    ovlnf_diff=abs(tovflc(idncl,swl-hdc(idncl,iseas),0,
     &               0.,iter,2))-abs(amin1(tovflc(idncl,swl_land_surf,0
     &               ,0.,iter,2),0.0))
                    ovlnf_diff=amax1(ovlnf_diff,0.0)
                    IF (nint(ovlnf_diff).gt.0) THEN
                      ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
                    ELSE
                      ovlnf_int=tovflc(idncl,swl-str_ds_stg_diff_frm_end
     &                ,0,0.,iter,2)
                    END IF
                  ELSE
                    ovlnf_diff=0.0
                    ovlnf_int=tovflc(idncl,swl-str_ds_stg_diff_frm_end
     &               ,0,0.,iter,2)
                  END IF
                ELSE
                  ovlnf_diff=0.0
                  ovlnf_int=tovflc(idncl,swl-str_ds_stg_diff_frm_end,0
     &            ,0.,iter,2)
                END IF
                cstor=(swl-chdep(idncl)-str_ds_stg_diff_frm_end)
     &           *cl(idncl)*width(idncl)
                cstor=cstor-cgseep(idncl,swl-str_ds_stg_diff_frm_end,0,
     &           0.)-abs(ovlnf_int)+ovlnf_diff-rfin(idncl)+etout(idncl)-
     &           qinfldn+coutlim*86400.
                cstor=amax1(cstor,0.0)/86400.
                surf_voldn=max(chdep(n)-mean_land_surf_along_canal,
     &           0.0)*gdar*nodcr(idncl)
                voldn=surf_voldn+voldn
                IF (cstor.lt.avail_cstor) THEN
                  avvoldn=cstor
                ELSE
```

```
                  avvoldn=avail_cstor+amax1(voldn/86400.,0.0)+
     &             amax1(coutlim-coutlim_ls,0.0)
                  avvoldn=amax1(avvoldn,0.0)
                END IF
              END IF
c             if (flnm(iflo).eq.'S381 ') print *,
c    &         coutfl,avvoldn,'  S381 '
              coutfl=amin1(coutfl,avvoldn)
            END IF
          END IF
          IF (flnm(istr(n,ind1)).eq.strname_spec(24)) THEN
            IF (trig_wca_stg(iwca3a).gt.regstg_wca(iwca3a,
     &       ibot_zone_indx(iwca3a+1))) THEN
              coutfl=0.0
            END IF
          END IF
          IF (ipurpse(n,ind1,ns).eq.2.and.irceive(n,ind1)
     &     .eq.1.and.idncl.le.nch) THEN
            IF (nout(n)+nstrctr(n).gt.1) THEN
              IF (nint(dqu(idncl)).gt.0) THEN
                coutfl=0.0
              END IF
            ELSE
              IF (conv.eq.1.) THEN
                outflow_to_ds_cnl1=amin1(coutfl,dqu(idncl))
                dqu(idncl)=dqu(idncl)-amin1(coutfl,dqu(idncl))
                icnl_dns1=idncl
   10           icnl_dns1=iup_canal_ws(icnl_dns1)
                IF (icnl_dns1.le.nch) THEN
                  dqu(icnl_dns1)=dqu(icnl_dns1)-outflow_to_ds_cnl1
                  dqu(icnl_dns1)=amax1(dqu(icnl_dns1),0.0)
                  GO TO 10
                END IF
              END IF
            END IF
          END IF
          IF (flnm(iflo).eq.strname_spec(79)) THEN
            CALL match (name_res_spec_code(14), 1, 6, resname, ntotres+
     &       1, 6, il8resbpit)
            coutfl=min(coutfl,volwtr(il8resbpit),
     &       max(weircff(il8resbpit,1,1)-tot_avl_l8_bpit_res_m1q-
     &       tot_avl_l8_bpit_res_lwd1-rl8runoff_to_bptres,0.0))
          END IF
          IF (.not.lflwtrg(ids_trg_idx(n,ind1))) THEN
            CALL match (flnm(iflo), 1, 6, struc_name_w_targ,
     &       no_of_struc_w_ds_targ_no_res, 6, istruc_targ_indx)
            coutfl=coutfl*frc_flow_targ(istruc_targ_indx)
          END IF
          IF (flnm(iflo).eq.strname_spec(84)) THEN
            coutfl1=coutfl-dischg_to_bay_s29_frm_c11r
            coutfl=coutfl1
          END IF
          IF (flnm(iflo).eq.'S334FC') THEN
            DO ncal = 1,nch
              DO i=1,nstrctr(ncal)
                IF (tmpname(ncal,i).eq.'S333  ') THEN
                   istruc_s333=i
```

```
                incl_s333 = ncal
              ENDIF
            END DO
          ENDDO
          dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
          IF (pond(node_g3273).gt.dpthgw) THEN
            stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
          ELSE
            stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
     &       ells(node_g3273)
          END IF
          IF (stage_g3273.le.stg_trigger_fl(incl_s333,istruc_s333
     &    ,1)) THEN
            IF (is334_open_flg.eq.0) THEN
              coutfl=amin1(coutfl*frac_avail_cap_s334,avvol_dns334)
            ELSE
              coutfl=amin1(coutfl,add_s333_fc_s334*
     &         frac_avail_cap_s334)
            END IF
          ELSE
            coutfl=add_s333_fc_s334
          END IF
          IF (nstorm_day(ndays_sim_sd).eq.1) coutfl=0.0
          s334_fc=coutfl
c          if (conv.eq.1.) print *,COUTFL,add_s333_fc_s334
c    $       ,is334_open_flg,avvol_dns334,s334_fc,'   S334FC  '
c    $       ,STAGE_G3273,stg_trigger_fl(ica3,istruc_s333,1)
        END IF
        IF (opt_fc_canal(n).eq.'FLWTH') avail_fc_vol=avail_fc_vol-
     $   coutf
        IF (imatch_struc_loc(n,ind1) .ne. 0 .and. irating(n,ind1,ns,
     $   imgt).eq.2 .and. iopt_for_interdep_on_off(n,ind1).eq.1) THEN
          if (i_counter_m(imatch_struc_loc(n,ind1)) .eq. 0) THEN
             rmax_dschg_c = discoef(n,ind1,1,1)
             isbsmx = 1
             IF (nsubstr(n,ind1) .gt. 1) THEN
                DO nss = 2,nsubstr(n,ind1)
                   IF (nint(discoef(n,i,nss,1)) .gt. rmax_dschg_c)
     $             rmax_dschg_c = discoef(n,i,nss,1)
                   isbsmx = nss
                ENDDO
             ENDIF
             insmx = ind1
             DO i = ind1+1,nstrctr(n)
               IF (imatch_struc_loc(n,i).eq.imatch_struc_loc(n,ind1))
     $          THEN
                i_counter_m(imatch_struc_loc(n,i))
     $          = i_counter_m(imatch_struc_loc(n,i)) + 1
                DO nss = 1,nsubstr(n,i)
                  IF (nint(discoef(n,i,nss,1)).gt.nint(rmax_dschg_c))
     $            THEN
                    rmax_dschg_c = discoef(n,i,nss,1)
                    insmx = i
                    isbsmx = nss
                  ENDIF
                ENDDO
               ENDIF
```

```
                ENDDO
               endif
               IF (flnm(iflo).eq. flnm(istr(n,insmx))) THEN
                dishg_lim_struc = 0.0
                DO i=1,nstrctr(n)
                  IF (imatch_struc_loc(n,i).eq.imatch_struc_loc(n,insmx))
     $            THEN
                     IF (i .ne. insmx) then
                       dishg_lim_struc = discoef(n,i,1,1)
                       insmn = i
                     ENDIF
                  ENDIF
                ENDDO
                IF (coutfl.lt.dishg_lim_struc) coutfl = 0.0
c    $          .and. stg_trig_fc_ops(n,ind1).gt.rmnstgs(n,insmn,iseas))
c    $          coutfl = 0.0
c               is_s381 = 0
c               DO iss = 1,nstrctr(ic11w)
c                 IF (flnm(istr(ic11w,iss)).eq.'S381 ') is_s381 = iss
c               ENDDO
c               IF (is_s381 .ne. 0) THEN
c                 IF (flnm(iflo).eq.'S9  '.and.nint(outf_fc_thru_s381)
c    $            .eq.0 .and. .not. lgrav(ic11w,is_s381,1)) coutfl = 0.0
c               ENDIF
c              if (conv.eq.1.)
c    $         print *,coutfl,dishg_lim_struc,stg_trig_fc_ops(n,ind1)
c    $         ,rmnstgs(n,insmn,iseas),'   S9_1'
               ENDIF
             ENDIF
            cout=coutfl+cout
c           if (cnm(n).eq.'NNRFG') print *,COUTFL,hw,tw,
c    $       rmnstgs(n,ind1,iseas),flnm(iflo),lgrav(n,ind1,ns),
c    $       lpump(n,ind1,ns),stg_trig_fc_ops(n,ind1)
c    $       ,rmnstgs(n,ind1,iseas)
c           if (conv.eq.1.and.flnm(iflo).eq.'S25B ') print *,COUTFL
c    $       ,'  S25BOUT ',chdep(n),hw,tw,rmxstg(n,ind1,iseas)
c    $       ,rmnstgs(n,ind1,iseas)
c           if (conv.eq.1.and.flnm(iflo).eq.'S335 ') print *,COUTFL
c    $       ,'  S335OT  ',chdep(n),chdep(l31nc)
            IF ((flnm(iflo).eq.strname_spec(66).or.flnm(iflo)
     &       .eq.strname_spec(79)).and.conv.eq.1.) THEN
              IF (flnm(iflo).eq.strname_spec(79)) THEN
                tot_avl_l8_bpit_res=0.0
              END IF
              tot_avl_l8_bpit_res=tot_avl_l8_bpit_res+coutfl
              IF (irating(n,ind1,ns,imgt).ne.2) THEN
                gravity_flow_struc=gravity_flow_struc+coutfl
              END IF
              IF (flnm(iflo).eq.strname_spec(66).and.conv.eq.1.) THEN
                tot_avl_l8_bpit_res_m1q=coutfl
              END IF
            END IF
            IF (flnm(iflo).eq.strname_spec(67).and.ns.eq.nsubstr(n,ind1)
     &       .and.conv.eq.1) THEN
              s331_pump=coutfl
            END IF
C
```

```
          IF (conv.eq.1) THEN
            IF (flnm(iflo).eq.strname_spec(21)) THEN
              IF (lok_split_option.eq.0) THEN
                finstor=finstor+cout*1.9835
                stagelo=stg2sto(finstor,1)
              ELSE
                finstor_res=finstor_res+cout*1.9835
                stagelo_res=stg2stores(finstor_res,1)
              END IF
            END IF
          END IF
C
        END IF
   20 CONTINUE
      IF (ispec(n,ind1).eq.0) THEN
C
        IF ((nint(wsdmnd).gt.0.or.flnm(iflo).eq.strname_spec(22)
     &   .or.flnm(iflo).eq.strname_spec(23)).and.ipurpse(n,ind1,1).ne.2)
     &    THEN
C
          IF (flnm(iflo).eq.strname_spec(22)) THEN
            IF (opt_flow_to_ts.eq.'TSMINDL') THEN
              wsdmnd=vol332(month)
            ELSE
              wsdmnd=vol332do(month)
            END IF
            wsdmnd=amax1(wsdmnd,0.0)
            avvol332=(chdep(n)-crmin(n,iseas))*cl(n)*width(n)+cgseep(n,
     &       crmin(n,iseas),0,0.)+tovflc(n,crmin(n,iseas),0,0.,iter,1)+
     &       qu(n)
            avvol332=amax1(avvol332,0.0)/86400.
            capacws(imgt)=amin1(discoef(n,ind1,1,1),avvol332)
          END IF
          IF (flnm(iflo).eq.strname_spec(23)) THEN
            wsdmnd=amax1(vol18c(month)/86400.,wsdmnd)
            wsdmnd=amax1(wsdmnd,0.0)
            avvol18c=(chdep(n)-crmin(n,iseas))*cl(n)*width(n)+cgseep(n,
     &       crmin(n,iseas),0,0.)+tovflc(n,crmin(n,iseas),0,0.,iter,1)+
     &       qu(n)
            avvol18c=amax1(avvol18c,0.0)/86400.
            capacws(imgt)=amin1(discoef(n,ind1,1,1),avvol18c)
          END IF
          rem_wsdmnd=wsdmnd-cout
          IF (conv.eq.1.) THEN
            IF (iserv_area_loc(idncl).gt.0) THEN
              rloc_excess_for_ws(iserv_area_loc(idncl))=
     &         rloc_excess_for_ws(iserv_area_loc(idncl))+amin1(cout/
     &         86400.,wsdmnd)*86400.
            END IF
          END IF
          rem_wsdmnd=amax1(rem_wsdmnd,0.0)
          IF (nprty(n,ind1).ne.0) THEN
            coutws=amin1(rem_wsdmnd,capacws(imgt)-cout,tempvol-cout)
            IF ((flnm(istr(n,ind1)).eq.strname_spec(7).or.flnm(istr(n,
     &       ind1)).eq.strname_spec(6)).and.conv.eq.1) THEN
              avvol_upstream=tempvol
            END IF
```

```
          ELSE
            IF (tdmnd.gt.0.0) THEN
              volrat=tempvol/tdmnd
            ELSE
              volrat=0.0
            END IF
            wratio(n)=amin1(volrat,1.0)
            wsneed=rem_wsdmnd*wratio(n)
            coutws=amin1(wsneed,capacws(imgt)-cout)
C
            tdmnd=tdmnd-wsdmnd
            IF ((flnm(istr(n,ind1)).eq.strname_spec(7).or.flnm(istr(n,
     &       ind1)).eq.strname_spec(6)).and.conv.eq.1) THEN
              avvol_upstream=volrat*wsdmnd
            END IF
          END IF
          IF (flnm(iflo).eq.'S334  ') coutws = min(capacws(imgt)-cout
     &     ,s333_ws)
          IF (conv.eq.1) THEN
            IF (flnm(iflo).eq.strname_spec(67)) s331_pump=amax1(coutws-
     &       tot_grav_cap,0.0)+s331_pump
          END IF
c           if (conv .eq.1.) then
c           if (cnm(n) .eq. 'L31N ') then
c              print *,tempvol,tdmnd,volrat
c    $        ,wratio(n),chdep(n),tw,tdqu_dn(idncl),cnm(idncl)
c    $        ,CAPACWS(IMGT),wsneed,coutws,'  S331WS'
c           endif
C           if (cnm(n) .eq. 'C9    ') then
C              print *,tempvol,tdmnd,volrat
C    $        ,wratio(n),chdep(n),chdep(idncl)+hdc(idncl,iseas)
C    $        ,CAPACWS(IMGT),wsneed,coutws,'  C9THLWD',coutws
C    $        ,cnm(idncl),dqu(idncl)
C           endif
c           endif
          tempvol=tempvol-amin1((coutws+cout),tempvol)
        ELSE
          coutws=0.0
          IF ((flnm(istr(n,ind1)).eq.strname_spec(7).or.flnm(istr(n,
     &     ind1)).eq.strname_spec(6)).and.conv.eq.1) THEN
            avvol_upstream=tempvol
          END IF
        END IF
        cout=cout+coutws
        IF (conv.eq.1.) THEN
          DO mm=1,nclns(idn_wca_indx(is8bsn-6),
     &     iconvey_canal_no(is150bws-6))
            IF (idncl.eq.isacnl(idn_wca_indx(is8bsn-6),
     &       iconvey_canal_no(is150bws-6),mm)) THEN
              outflow_from_3a(iconvey_canal_no(is150bws-6))=
     &         outflow_from_3a(iconvey_canal_no(is150bws-6))+cout
            END IF
          END DO
        END IF
        IF (flnm(istr(n,ind1)).eq.strname_spec(67).and.conv.eq.1.) THEN
          s331ws=amin1(cout,wsdmnd)
          s331fc=cout-s331ws
```

```
        END IF
        IF (flnm(istr(n,ind1)).eq.strname_spec(7)
     &   .and.n.eq.intcnl(idn_wca_indx(is8bsn-6),
     &   iconvey_canal_no(is150bws-6))) THEN
          IF (conv.eq.1.) THEN
            outflow_from_3a(iconvey_canal_no(is150bws-6))=
     &       outflow_from_3a(iconvey_canal_no(is150bws-6))+cout
            outflow_from_3a_temp(iconvey_canal_no(is150bws-6))=
     &       outflow_from_3a(iconvey_canal_no(is150bws-6))
          ELSE
            outflow_from_3a_temp(iconvey_canal_no(is150bws-6))=
     &       outflow_from_3a(iconvey_canal_no(is150bws-6))+cout
          END IF
        END IF
        IF ((flnm(istr(n,ind1)).eq.strname_spec(7).or.flnm(istr(n,ind1))
     &   .eq.strname_spec(6)).and.conv.eq.1) THEN
          IF (lp(23)) THEN
            WRITE (62,30) ifyr+iyear-1,month,iday,flnm(istr(n,ind1)),hw,
     &      tw,capacws(imgt)+capacfl(imgt,ns),avvol_upstream,cout,wsdmnd
          END IF
   30     FORMAT (i5,2i3,5x,a6,5x,2f7.2,8f10.0)
        END IF
C
C  CALCULATE MAXIMUM AVAILABLE STORAGE IN RESERVOIR RECEIVING DISCHARGE
C   FROM STRUCTURE,IF APPLICABLE
C
        diffdrs=0.0
        coutrs=cout
        tcoutrs=0.0
        IF (nres(n,ind1).gt.0) THEN
          DO ir=1,nres(n,ind1)
            IF ((.not.lwslo(itrg_byp_r_idx(n,ind1,ir))
     &       .and.iopt_to_deiver_bel_targ(n,ind1,ir).eq.0)
     &       .or.(lwslo(itrg_byp_r_idx(n,ind1,ir))
     &       .and.iopt_to_deiver_bel_targ(n,ind1,ir).eq.1)) THEN
              couttp=pctres(n,ind1,ir)*cout+diffdrs
              coutrs=amin1(couttp,tivolwtr(irsrcvr(n,ind1,ir)))
              diffdrs=cout*pctres(n,ind1,ir)-coutrs
c             if (resname(IRSRCVR(N,IND1,IR)) .eq. 'C4IMP '.and.conv
c    &        .eq.1) print *, COUTRS,VOLWTR(IRSRCVR(N,IND1,IR))
c    &          ,'  IN WROUTE',POND(NODRCV(N,IND1,IR)),
c    &          tivolwtr(irsrcvr(n,ind1,ir))
              tivolwtr(irsrcvr(n,ind1,ir))=tivolwtr(irsrcvr(n,ind1,ir))-
     &         coutrs
              IF (conv.eq.1.) THEN
c             if (resname(IRSRCVR(N,IND1,IR)) .eq. 'C4IMP ')
c    &          print *, COUTRS,VOLWTR(IRSRCVR(N,IND1,IR))
c    &          ,'  IN WROUTE',POND(NODRCV(N,IND1,IR)),
c    &          tivolwtr(irsrcvr(n,ind1,ir))
              volwtr(irsrcvr(n,ind1,ir))=volwtr(irsrcvr(n,ind1,ir))-
     &          coutrs
              rinflw_to_res_frm_cnl(irsrcvr(n,ind1,ir))=
     &          rinflw_to_res_frm_cnl(irsrcvr(n,ind1,ir))+coutrs
              daily_inflow_to_res_frm_cnl(irsrcvr(n,ind1,ir))=coutrs+
     &          daily_inflow_to_res_frm_cnl(irsrcvr(n,ind1,ir))
              nstrtores(irsrcvr(n,ind1,ir))=nstrtores(irsrcvr(n,ind1,
     &          ir))+1
```

```
                irfsta=irsrcvr(n,ind1,ir)
                IF (irceive(n,ind1).eq.2) THEN
                  dpthgw=(ells(nodrcv(n,ind1,ir))-h(nodrcv(n,ind1,ir)))/
     &             s(nodrcv(n,ind1,ir))
                  IF (pond(nodrcv(n,ind1,ir)).gt.dpthgw) THEN
                    est_stage=pond(nodrcv(n,ind1,ir))-dpthgw+
     &               ells(nodrcv(n,ind1,ir))
                  ELSE
                    est_stage=(pond(nodrcv(n,ind1,ir))-dpthgw)/
     &               s(nodrcv(n,ind1,ir))+ells(nodrcv(n,ind1,ir))
                  END IF
                  gw_stor_res_cell=amax1(ells(nodrcv(n,ind1,ir))-
     &             est_stage,0.0)*gdar/86400.
                  est_flow_to_gwt(irfsta)=est_flow_to_gwt(irfsta)+
     &             amin1(coutrs,gw_stor_res_cell)
                  IF (sfactor(irfsta)
     &             .ge.sfactmin.or.ires_small_sim(irfsta).eq.'NO ')
     &             THEN
                    pond(nodrcv(n,ind1,ir))=pond(nodrcv(n,ind1,ir))+
     &               coutrs*86400./(gdar*(1.0-carfact(nodrcv
     &               (n,ind1,ir))-total_small_afact(nodrcv(n,ind1,ir))))
                    IF (resname(irfsta).eq.'STA1E ') THEN
                      res_struc_inflw(irsrcvr(n,ind1,ir))=
     &                 res_struc_inflw(irsrcvr(n,ind1,ir))+coutrs*
     &                 1.9835
                    END IF
                  ELSE
                    DO kk=1,noresincell(nodrcv(n,ind1,ir))
                      IF (irsrcvr(n,ind1,ir)
     &                 .eq.ires_index_in_cell(nodrcv(n,ind1,ir),kk))
     &                 THEN
                       IF (stage_res(nodrcv(n,ind1,ir),kk)
     &                  .gt.ells_in_res(nodrcv(n,ind1,ir),kk)) THEN
                        pond_in_res(nodrcv(n,ind1,ir),kk)=
     &                   pond_in_res(nodrcv(n,ind1,ir),kk)+coutrs*
     &                   86400./(sfactor(irsrcvr(n,ind1,ir))*gdar)
                        stage_res(nodrcv(n,ind1,ir),kk)=
     &                   ells_in_res(nodrcv(n,ind1,ir),kk)+
     &                   pond_in_res(nodrcv(n,ind1,ir),kk)
                       ELSE
                        dpth_below_ls_res=(ells_in_res(nodrcv(n,ind1,
     &                   ir),kk)-stage_res(nodrcv(n,ind1,ir),kk))*
     &                   s(nodrcv(n,ind1,ir))
                        dpth_of_inflw=coutrs*86400./
     &                   (sfactor(irsrcvr(n,ind1,ir))*gdar)
                        stage_res(nodrcv(n,ind1,ir),kk)=
     &                   stage_res(nodrcv(n,ind1,ir),kk)+
     &                   amin1(dpth_of_inflw,dpth_below_ls_res)/
     &                   s(nodrcv(n,ind1,ir))+amax1(dpth_of_inflw-
     &                   dpth_below_ls_res,0.0)
                        pond_in_res(nodrcv(n,ind1,ir),kk)=
     &                   amax1(stage_res(nodrcv(n,ind1,ir),kk)-
     &                   ells_in_res(nodrcv(n,ind1,ir),kk),0.0)
                       END IF
C                      if (resname(IRSRCVR(N,IND1,IR)) .eq.'C9RES ')
C    &                  print *,pond_in_res(NODRCV(N,IND1,IR),kk)
C    &                  ,stage_res(NODRCV(N,IND1,IR),kk),COUTRS
```

```
C    &                  ,ells_in_res(NODRCV(N,IND1,IR),kk)
C    &                  ,'  PLLYPLLYPLLY  ',NODRCV(N,IND1,IR)
                       res_struc_inflw(irsrcvr(n,ind1,ir))=
     &                  res_struc_inflw(irsrcvr(n,ind1,ir))+coutrs*
     &                  1.9835
                      END IF
                    END DO
                  END IF
                ELSE
                  qu(idncnl(n,ind1))=qu(idncnl(n,ind1))+coutrs*86400.
                END IF
c         if (conv.eq.1)then
c         if (cnm(n).eq.'C4   ') print *,CHDEP(N),COUT
c    &     ,IBSN(NODRCV(N,IND1,1)),VOLWTR(IRSRCVR(N,IND1,1))
c    &     ,IRSRCVR(N,IND1,1),COUTRS,COUTTP,' MOCNLTORES  '
c    &     ,POND(NODRCV(N,IND1,IR))
c         endif
              END IF
            ELSE
              coutrs=0.0
            END IF
            tcoutrs=tcoutrs+coutrs
          END DO
          DO ir=1,nres(n,ind1)
            IF (pctres(n,ind1,ir).lt.0.010) THEN
              icres_dn=0
              DO j=1,nstrctr(idncl)
                IF (nres(idncl,j).gt.0) THEN
                  DO irr=1,nres(idncl,j)
                    IF (irsrcvr(idncl,j,irr).eq.irsrcvr(n,ind1,ir))
     &               THEN
                      icres_dn=icres_dn+1
                    END IF
                  END DO
                END IF
              END DO
              IF (icres_dn.ne.0) THEN
                IF (resname(irsrcvr(n,ind1,ir))
     &           .eq.name_res_spec_code(15).or.resname(irsrcvr(n,ind1,
     &           ir)).eq.name_res_spec_code(18)) THEN
                  temp_availstor=temp_volwtr(irsrcvr(n,ind1,ir))-
     &             outf_from_c11res
                  temp_availstor=amax1(temp_availstor,0.0)
                  temp_struccap=temp_weircff(irsrcvr(n,ind1,ir),
     &             i_res_inf_struc_indx(n,ind1,ir))-outf_from_c11res
                ELSE
                  temp_availstor=temp_volwtr(irsrcvr(n,ind1,ir))
                  temp_struccap=temp_weircff(irsrcvr(n,ind1,ir),
     &             i_res_inf_struc_indx(n,ind1,ir))
                END IF
                cout=amin1(cout,temp_availstor,temp_struccap)
                IF (conv.eq.1.) THEN
                  temp_volwtr(irsrcvr(n,ind1,ir))=temp_volwtr(irsrcvr(n,
     &             ind1,ir))-cout/nres(n,ind1)
                  temp_weircff(irsrcvr(n,ind1,ir),
     &             i_res_inf_struc_indx(n,ind1,ir))=
     &             temp_weircff(irsrcvr(n,ind1,ir),
```

```
&             i_res_inf_struc_indx(n,ind1,ir))-cout/nres(n,ind1)
            END IF
          ELSE
            IF (resname(irsrcvr(n,ind1,ir))
&            .eq.name_res_spec_code(15).or.resname(irsrcvr(n,ind1,
&            ir)).eq.name_res_spec_code(18)) THEN
              temp_availstor=volwtr(irsrcvr(n,ind1,ir))-
&              outf_from_c11res
              temp_availstor=amax1(temp_availstor,0.0)
            ELSE
              temp_availstor=volwtr(irsrcvr(n,ind1,ir))
            END IF
            cout=amin1(cout,temp_availstor)
          END IF
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              upstream_inflow(idncl)=upstream_inflow(idncl)+coutws*
&              86400.
              dqu(idncl)=dqu(idncl)-amin1(coutws,dqu(idncl))
            ELSE IF (irceive(n,ind1).eq.2) THEN
              DO j=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,j))=pond(nodrcv(n,ind1,j))+cout*
&                86400/(ncells_rec(n,ind1)*gdar*(1.0-
&                total_small_afact(nodrcv(n,ind1,j))))
              END DO
            ELSE IF (irceive(n,ind1).eq.3.and.irvbsn(n,ind1).ne.-1)
&            THEN
              DO iy=maxy,1,-1
                DO node=isum(iy)+1,isum(iy+1)
                  IF (ibsn(node).eq.irvbsn(n,ind1)) THEN
                    pond(node)=pond(node)+cout*86400./
&                    (nndrec(ibsn(node))*gdar*(1.0-
&                    total_small_afact(node)))
                  END IF
                END DO
              END DO
            END IF
            IF (flnm(iflo).eq.strname_spec(89))
&            outf_to_c9res_via_us27s=cout
          END IF
        ELSE
          cout=tcoutrs
        END IF
      END DO
    ELSE
      IF (conv.eq.1.) THEN
        IF (irceive(n,ind1).eq.1) THEN
          qu(idncl)=qu(idncl)+cout*86400.
          upstream_inflow(idncl)=upstream_inflow(idncl)+coutws*
&          86400.
          dqu(idncl)=dqu(idncl)-amin1(coutws,dqu(idncl))
          IF (idncl.lt.n) THEN
            rloc_excss_to_canl_next_step(idncl)=
&            rloc_excss_to_canl_next_step(idncl)+cout*86400.
          END IF
        ELSE IF (irceive(n,ind1).eq.2) THEN
```

```
              DO j=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,j))=pond(nodrcv(n,ind1,j))+cout*
     &           86400/(ncells_rec(n,ind1)*gdar*(1.0-
     &           total_small_afact(nodrcv(n,ind1,j))))
              END DO
            ELSE IF (irceive(n,ind1).eq.3.and.irvbsn(n,ind1).ne.-1)
     &       THEN
              DO iy=maxy,1,-1
                DO node=isum(iy)+1,isum(iy+1)
                  IF (ibsn(node).eq.irvbsn(n,ind1)) THEN
                    pond(node)=pond(node)+cout*86400./(nndrec(ibsn(node)
     &               )*gdar*(1.0-total_small_afact(node)-carfact(node)))
                  END IF
                END DO
              END DO
            END IF
          END IF
        END IF
        IF (tmpname(n,ind1).eq.'L8TSTA') THEN
          flow_to_l8sta=cout
        END IF
        IF (flnm(istr(n,ind1)).eq.'S9   ') THEN
          outf_s9 = cout
        ENDIF
        IF (flnm(istr(n,ind1)).eq.'S9A  ') THEN
          outf_s9a = cout
        ENDIF
C
C  SAVE OUTFLOW FOR G-123 FOR USE IN CALCULATION OF FLOW FOR
C     S-142 AND S-34 IN WACOUT SUBROUTINE
C
        IF (conv.eq.1) THEN
          IF (flnm(istr(n,ind1)).eq.strname_spec(24)) THEN
            tflg123=cout
          END IF
          IF (flnm(istr(n,ind1)).eq.strname_spec(25)) THEN
            ws_to_c6w=cout
          END IF
          IF (flnm(iflo).eq.strname_spec(14)) THEN
            s336_to_biscbay=cout
          END IF
          IF (flnm(istr(n,ind1)).eq.strname_spec(88)) THEN
            outflow_from_us27s=cout
          END IF
          IF (flnm(istr(n,ind1)).eq.strname_spec(90)) THEN
            outflow_from_us27n=cout
          END IF
          IF (flnm(istr(n,ind1)).eq. 'S381 ') THEN
            outf_fc_thru_s381 = cout
          ENDIF
C
          IF (flnm(istr(n,ind1)).eq.strname_spec(1)) THEN
            kflo(istr(n,ind1))=cout+kflo(istr(n,ind1))
          ELSE IF (flnm(istr(n,ind1)).eq.strname_spec(13)) THEN
            IF (cnm(n).eq.canl_names_spec(27)) THEN
              kflo(istr(n,ind1))=cout
            ELSE IF (cnm(n).eq.canl_names_spec(32)) THEN
```

```
          kflo(istr(n,ind1))=-cout
        END IF
      ELSE
        kflo(istr(n,ind1))=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.strname_spec(9)) THEN
        flow_s155=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'LBFC11') THEN
        flow_to_lbn_frm_c11=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'C9RC11') THEN
        flow_to_c9r_frm_c11=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'E1TPBR') THEN
        flow_to_cpbres_frm_e1=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'ACMRIN') THEN
        flow_to_acmres_frm_basin=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'C9TC9R') THEN
        flow_to_c9r_frm_c9=cout
      END IF
      IF (flnm(istr(n,ind1)).eq.'ACBCA1') THEN
        flow_to_ca1_frm_acmebsn=cout
      END IF
      IF (cnm(n).eq.'LBLTN') tot_flw_to_c9r_frm_nlkblt=
&      tot_flw_to_c9r_frm_nlkblt+cout
    END IF
  END IF
  RETURN
  END
```

gen_model_def_param.F

```
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: gen_model_def_param.F,v 1.30 2005/06/10 13:29:06 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/gen_model_def_param.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO RETRIEVE GENERAL DEFINITION DATA FOR MODEL DOMAIN
C   FILE OPENED: model_definition_data.info.dat
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE gen_model_def_param_data (ifacteaa, isem_flg,
     & max_days_mean_loktasr, max_days_mean_bkpump, rf_factor,
     & ro_factor, capac_sugh, lok_targ_level, use_ssm, import_area_name,
     & ip, npopts, nsvarea, nlu, capacc10a_bak, desgn_cap_canl_ws,
     & welldat_opt, ntstrctr, env_ws_cutb_ssm_opt, bmp_opt,
     & opt_bsn_prio_uncond, opt_prior_use_asr_flex_cal,
     & frac_thru_s8_sta34, n_cells, res_seep_factor, ds_convey_c_name,
     & ieaa_res_asr_name, ieaa_res_asr_reg_name, name_res_for_inflow,
     & eaa_conv_canal_names, base_area_eaa_basin, idn_wca_name,
     & idn_wca_name_env, depth_soil_eaa, solinit, icol_stg, irow_stg,
     & int_cnl_name_for_lwdd, ds_int_canal_name_l8, idn_wca_name_l8,
     & idowns_wca_name_lwdd, fracdph_min, fracdph_max, ssmminfrac,
     & lok_targ_level_wet1, lok_targ_level_wet2, issm_cutb_opt_bcyp,
     & issm_cutb_opt_brghton, rlok_stg_beg_forpmp,
     & rlok_stg_end_forpmp, iregcals79opt, ws_bkp_cap, maxbpcntr,
     & rf_factor_lok, ro_factor_lok, nwgoalsto, mon_targ_wet,
     & iday_targ_wet, struc_name_meas, iwsbkpwsta, bflo_frac_c43est,
     & bflo_cap_l8, frac_dmnd_met_wt, frac_dmnd_met_wn,
     & rmean_cnl_bot_elev, flow_resistvty_coef, rf_drawdwn_dph,
     & hdrop_fact, hdrop_max, no_canl_in_cell, icanl_no_in_cell,
     & ipos_cell_in_canl, rlosa_factor, s235_capac, s235_frac,
     & diston_capac, diston_frac, semirr_dmnd_factor, fpl_dmnd_factor,
     &hdrop_min,bmp_ssm_opt,pcbruleopt,pcbfrac,frac_cal_dmnd_met_by_del,
```

```
&allig_alley_res_fact,iroute_l8rnf_to_sta_opt)

 INCLUDE 'wmm_par.inc'
 INCLUDE 'abc.inc'
 INCLUDE 'agdata.inc'
 INCLUDE 'c1.inc'
 INCLUDE 'c2.inc'
 INCLUDE 'et.inc'
 INCLUDE 'stat.inc'
 INCLUDE 'wrout.inc'
 INCLUDE 'routc.inc'
 INCLUDE 'stas.inc'
 INCLUDE 'simflows.inc'
 INCLUDE 'wcaot.inc'
 INCLUDE 'lakwca.inc'
 INCLUDE 'environ.inc'
 INCLUDE 'simcstgs.inc'
 INCLUDE 'wier.inc'
 INCLUDE 'trig_lec_et_param.inc'
 INCLUDE 'trigger.inc'
 INCLUDE 'lec_et.inc'
 INCLUDE 'eaacap.inc'
 INCLUDE 'mgateop.inc'
 INCLUDE 'opersched.inc'
 INCLUDE 'STRCOUNT.inc'
 INCLUDE 'daydump.inc'
 INCLUDE 'loktarg.inc'
 INCLUDE 'wcasim.inc'
 INCLUDE 'canalloc.inc'
 INCLUDE 'reuse.inc'
 INCLUDE 'lokres.inc'
 INCLUDE 'dlycslope.inc'

 DIMENSION base_area_eaa_basin(max_n_eaa_condt),
& desgn_cap_canl_ws(max_n_eaa_condt), detenc_canl(max_n_lu_type),
& fracdph_max(12,3), fracdph_min(12,3),no_canl_in_cell(max_ncells),
& hdrop_fact(max_n_cnls_w_var_slpe),
& hdrop_max(max_n_cnls_w_var_slpe), icanal_indx_lwdd(10),
& icol_cell_wca_daily(max_n_targ_area,max_n_targ_loc),
& icol_cell_wca_daily_out(max_n_targ_area,max_n_targ_loc),
& icol_reuse(max_n_reuse_plnts),icanl_no_in_cell(max_ncells,3),
& icol_stg(max_n_targ_area,max_n_targ_loc),
& iday_targ_wet(max_n_goal_ssm), ieaa_str_indx(max_n_eaa_condt,5),
& ip(60), irow_cell_wca_daily(max_n_targ_area,max_n_targ_loc),
& irow_cell_wca_daily_out(max_n_targ_area,max_n_targ_loc),
& irow_reuse(max_n_reuse_plnts),hdrop_min(max_n_cnls_w_var_slpe),
& irow_stg(max_n_targ_area,max_n_targ_loc), istr_indx_lwdd(10),
& iup_canal_indx_lwdd(10), ix(max_n_gage_loc_outp),
& ixdnslope(max_n_cnls_w_var_slpe),ipos_cell_in_canl(max_ncells,3),
& ixupslope(max_n_cnls_w_var_slpe), iy(max_n_gage_loc_outp),
& iydnslope(max_n_cnls_w_var_slpe),
& iyupslope(max_n_cnls_w_var_slpe), l8basin_canal_indx(5),
& mon_targ_wet(max_n_goal_ssm), n_cells(max_n_gage_loc_outp),
& n_eaa_str(max_n_eaa_condt), n_offset_bkpts(max_n_targ_area),
& ofmc_canl(max_n_lu_type,4),
& rf_drawdwn_dph(max_ncnls,2,max_n_seas),
& rf_drawdwn_dph0(2,max_n_seas), rf_factor(12), rf_factor_lok(12),
```

```
     & rlok_stg_beg_forpmp(max_n_eaa_condt), rlosa_factor(8,2),
     & rlok_stg_end_forpmp(max_n_eaa_condt), ro_factor(12),
     & ro_factor_lok(12), ws_bkp_cap(max_n_eaa_condt,2),
     & diston_capac(2),diston_frac(2),pcbfrac(4)

      CHARACTER bmp_opt*5, canal_lwdd_name(10)*5,
     & canal_name_output(max_ncnls)*5,icanal_name_ssm(max_ncnls)*5,
     & canal_reuse_names(max_n_reuse_plnts)*5, canl_names_sw_int*5,
     & cistc(max_n_strt_cnls_ws)*5,
     & cnl_fc_flwth_name(max_n_flwth_cnls)*5,
     & cnl_flwth_name(max_n_flwth_cnls)*5, days_week(7)*3,
     & days_week_del_eaa_ssm(7,2)*3,
     & ds_convey_c_name(max_n_eaa_condt)*5, ds_int_canal_name_l8*5,
     & eaa_conv_canal_names(max_n_eaa_condt)*5, eaa_str_name(10)*6,
     & env_ws_cutb_ssm_opt*3, icnl_dn_s31_name*5,use_ssm*3,
     & idn_wca_name(max_n_eaa_condt)*5, bmp_ssm_opt*3,
     & idn_wca_name_env(max_n_eaa_condt)*5,
     & idn_wca_name_l8*5, str_lwdd_name(10)*6, idowns_wca_name_lwdd*5,
     & ieaa_res_asr_name(max_n_eaa_condt)*6,
     & ieaa_res_asr_reg_name(max_n_eaa_condt,max_n_reg_rel_basin)*6,
     & import_area_name(max_n_targ_area)*5,
     & int_cnl_holey_name(max_n_eaa_condt)*5,
     & int_cnl_name_for_lwdd*5, iup_canals_lwdd_name(10)*5,
     & l8basin_canal_name(10)*5,
     & name_res_for_inflow(max_n_div_to_ecp)*6, ncl_ws_name(30)*5,
     & nds_cnl_ws_name(30)*5, opt_bsn_prio_uncond*3,
     & opt_prior_use_asr_flex_cal*3, sa_canal_name(max_n_ups_cnl_sa)*5,
     & struc_name_meas(max_n_struc_meas)*6,
     & targ_area_name(max_n_eaa_condt)*5, welldat_opt*5,pcbruleopt*3,
     & name_eaa_cond_298_s(max_n_eaa_condt)*5

      REAL flow_resistvty_coef(max_n_cnls_w_var_slpe), lok_targ_level,
     & lok_targ_level_wet1(max_n_goal_ssm),
     & lok_targ_level_wet2(max_n_goal_ssm),
     & rmean_cnl_bot_elev(max_n_cnls_w_var_slpe), ssmminfrac

      DATA days_week /'SUN', 'MON', 'TUE', 'WED', 'THU', 'FRI', 'SAT'/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  SET UP CANAL NODE LENGTHS

      DISTNC(1) = DX
      DISTNC(2) = DY
      DISTNC(3) = SQRT ( DX * DX  +  DY * DY)
      GDAR = DX * DY
C
      DO 41 I=1,4
         AGP(I) = 0.
   41    BAGP(I) = 0.
c
      is8bsn = 7
      is7bsn = 8
      is6bsn = 10
      is5absn = 9
      is150bws = 11
      il8bsn = 11
```

```
c
      lsim_s65e_clim = .false.
c
      do node = 1,max_ncells
         no_canl_in_cell(node) = 0
      enddo
c
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C  INPUT NUMBER OF CANALS SIMULATED,CANAL LOCATIONS
c
      NPTS = 0
      NSTEP = 1
      N = 0
 255  N = N + 1
      CL(N) = 0
      READ (23,'(A5,I3)') cnm(n),nodcr(n)
      IF (cnm(n) .eq. 'NOCNL') then
         nch = n - 1
         GOTO 270
      ENDIF
      DO ivars = 11,55,11
        IF (nodcr(n) .gt. ivars) THEN
          READ (23,260) (xcn(n,ic), ycn(n,ic), icl(n,ic), ic=ivars-10,
     &      ivars)
        ELSE
          icount = nodcr(n)
          READ (23,260) (xcn(n,ic), ycn(n,ic), icl(n,ic), ic=ivars-10,
     &      icount)
          GOTO 265
        ENDIF
      ENDDO
 260  FORMAT (5X,11(2I3,I2,1X))
C-----------------
C  STORE ALL CANAL NODES IN NODECN ARRAY.
C      NPTS KEEPS TRACK OF WHERE THEY ARE.
C-----------------
 265  DO 4 I = 1,NODCR(N)
        NPTS = NPTS + 1
        NODECN(NPTS) = (XCN(N,I) - MINX(YCN(N,I)) + 1 + ISUM(YCN(N,I)))
     &    * 10 + ICL(N,I)
        icell_no = NODECN(NPTS)/10
        CL(N) = CL(N) + DISTNC(ICL(N,I))
        no_canl_in_cell(icell_no) = no_canl_in_cell(icell_no) + 1
        icanl_no_in_cell(icell_no,no_canl_in_cell(icell_no)) = n
        ipos_cell_in_canl(icell_no,no_canl_in_cell(icell_no)) = i
   4   CONTINUE
       ivary_slope_daily_opt(n) = 0
       icanal_swint_dep(n) = 0
       mid_point_canl(N) = NODCR(N)/2
       GOTO 255
 270   read (23,'(i3,2x,17(A5,1x),3(/5x,17(A5,1x)))') no_canl_names_spec
     $   ,(canl_names_spec(i),i=1,no_canl_names_spec)
c
c  input surface water interaction data for canals input here
c
       read (23,*) no_canl_sw_int
```

```
      do nc = 1,no_canl_sw_int
         read (23,*) canl_names_sw_int,(ofmc_canl(nc,i)
     $     ,i=1,4),detenc_canl(nc)
         call match (canl_names_sw_int,1,5,cnm,nch,5,imatch)
         if (imatch .eq. 0) then
            write(*,2027) canl_names_sw_int
 2027       format(2(/),'UNRECOGNIZED CANAL NAME',2x,A6,2x,'INPUT FOR U
     $NIQUE SW INTERACTION DATA. CHECK CANAL_GRID_LOC FILE')
           stop
         endif
         icanal_swint_dep(imatch) = nc
      enddo
c
c
      do n = 1,nch
        if (icanal_swint_dep(n) .ne. 0) then
          do ic = 1,nodcr(n)
            do jj = 1,4
              ofm_canl(n,ic,jj) = ofmc_canl(icanal_swint_dep(n),jj)
            enddo
            deten_cnl(n,ic) = detenc_canl(icanal_swint_dep(n))
          enddo
        endif
      enddo
c
c  input which canals whose demands will be cutback according to SSM
c
      read (23,*) no_canals_ssm,(icanal_name_ssm(i),i=1,no_canals_ssm)
      do i=1,no_canals_ssm
         call match (icanal_name_ssm(i),1,5,cnm,nch,5,imatch)
         IF (imatch .eq. 0) THEN
            write(*,3037) icanal_name_ssm(i)
 3037       format(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,'INPUT FOR S
     $SM CUTBACK. CHECK CANAL_GRID_LOC FILE')
            stop
         ENDIF
         icanal_ssm(i) = imatch
      enddo
c
c  input data for canals whose head drop will vary daily
c
c
      kk = 1
 256  read (23,'(A5,2x,2f7.2,4i5,7F5.1)',end = 257) canal_name_dvslope
     $  (kk),rmean_cnl_bot_elev(kk),flow_resistvty_coef(kk)
     $  ,ixupslope(kk),iyupslope(kk),ixdnslope(kk),iydnslope(kk)
     $  ,(rf_drawdwn_dph0(1,ll),ll=1,2),(rf_drawdwn_dph0(2,ll),ll=1,2)
     $  ,hdrop_fact(kk),hdrop_max(kk),hdrop_min(kk)
      call match (canal_name_dvslope(kk),1,5,cnm,nch+1,5,imatch_v)
      if (imatch_v .eq. 0) then
         write(*,2028) canal_name_dvslope(kk)
 2028    format(2(/),'UNRECOGNIZED CANAL NAME',2x,A6,2x,'INPUT FOR DAIL
     $Y VARYING HEAD DROP. CHECK CANAL_GRID_LOC FILE')
         stop
      endif
      ivary_slope_daily_opt(imatch_v) = 1
      do ll = 1,2
```

```
          rf_drawdwn_dph(imatch_v,1,ll) = rf_drawdwn_dph0(1,ll)
          rf_drawdwn_dph(imatch_v,2,ll) = rf_drawdwn_dph0(2,ll)
        enddo
        kk = kk + 1
        go to 256
 257    continue
        ncnl_dly_slope = kk-1
        do ll = 1,ncnl_dly_slope
          if (ixupslope(ll) .gt. 0 .and. ixdnslope(ll) .gt. 0) then
             igrid_loc_upslp(ll) = ixupslope(ll) - minx(iyupslope(ll))
     $       + 1 + isum(iyupslope(ll))
             igrid_loc_dnslp(ll) = ixdnslope(ll) - minx(iydnslope(ll))
     $       + 1 + isum(iydnslope(ll))
          else
             igrid_loc_upslp(ll) = -901
             igrid_loc_dnslp(ll) = -901
          endif
        enddo
C
C  read in header information for daily import file for wcas
c
         if (wca_import_variation .ne. 'DAILY') then
          read (105,*) ntotal_no_import_areas_const
          do isa = 1,ntotal_no_import_areas_const
           read (105,*) import_area_name(isa),nmpts_wca(isa)
     $      ,(icol_stg(isa,i),irow_stg(isa,i),rimport_stage(isa,i)
     $      ,i=1,nmpts_wca(isa))
c
          enddo
          read (105,*) import_area_name(ntotal_no_import_areas_const+1)
         endif
c
         read (105,'(2i5)')ngrid_cells_daily_total,ntotal_no_import_areas
          do isa = 1,ntotal_no_import_areas
             read (105,'(A5,2x,i5,2f6.2,50i5)') import_area_name(isa),
     $       n_offset_bkpts(isa),( offset(isa,i),i=1,n_offset_bkpts
     $       (isa)),ngrid_cells_daily(isa),(ngage_index_env(isa,i,1),
     $       icol_cell_wca_daily(isa,i),irow_cell_wca_daily
     $       (isa,i),i=1,ngrid_cells_daily(isa))
             read (105,'(12x,2f6.2,50i5)') ( offset_out(isa,i),
     $       i=1,n_offset_bkpts(isa)),ngrid_cells_daily_out(isa),
     $       (ngage_index_env(isa,i,2),icol_cell_wca_daily_out(isa
     $       ,i),irow_cell_wca_daily_out(isa,i),i=1,ngrid_cells_daily_out
     $       (isa))
          enddo
          if (wca_import_variation .eq. 'DAILY') then
           read (105,'(A5)') import_area_name(ntotal_no_import_areas+1)
          endif
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
          do isa = 1,ntotal_no_import_areas
           if (ngrid_cells_daily(isa) .gt. 0) then
            do i = 1,ngrid_cells_daily(isa)
             node_import_daily(isa,i) = icol_cell_wca_daily(isa,i)
     $       -minx(irow_cell_wca_daily(isa,i)) + 1 +
     $       isum(irow_cell_wca_daily(isa,i))
            enddo
           endif
```

```
          if (ngrid_cells_daily_out(isa) .gt. 0) then
           do i = 1,ngrid_cells_daily_out(isa)
            node_outflw_daily(isa,i) = icol_cell_wca_daily_out(isa,i)
     $       -minx(irow_cell_wca_daily_out(isa,i)) + 1 +
     $       isum(irow_cell_wca_daily_out(isa,i))
           enddo
          endif
         enddo
c
c  general model run definition parameters
c
      read (2,'(A5)') welldat_opt
      read (2,*) (rf_factor(i),i=1,12)
      read (2,*) (ro_factor(i),i=1,12)
      read (2,*) (rf_factor_lok(i),i=1,12)
      read (2,*) (ro_factor_lok(i),i=1,12)
      read (2,*) proj_sea_level_rise
      read (2,'(I4)') netzon
      read (2,'(I4)') nlu
c
c WRMI, 1/5/93, change for trigger module
      if(use_trigger .ge. 1) then
          max_row = maxy
          do i=1,maxy
              max_column = max0(max_column, maxx(i))
          enddo
      endif
c
c  read names of structures with measured flows in simulation
c
      READ (2,'(I3,2x,60(A6,1x))') nflpts,(struc_name_meas(i)
     $  ,i=1,nflpts)
c
c  read names of canals desired to be output
c
      nmc = nch
      DTS = DT / NTSTEP_OV * 86400.
c
      do i = 1,nmc
        canal_name_output(i) = cnm(i)
        mtrc(i) = i
        cnlname(i) = canal_name_output(i)
      enddo
      maxstrc = nmc
      maxstrc = maxstrc + 1
      cnlname(maxstrc) = 'LOK  '
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c  read master list of names of structures as basis for indexing
c
      READ (2,35) NTSTRCTR,(FLNM(I),I=1,NTSTRCTR)
 35   FORMAT(I3,2x,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/
     $  5X,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/5X,
     $  20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/5X,20(A6,1x)/
     $  5X,20(A6,1x)/5X,20(A6,1x))
c
      call match('S5A2  ',1,6,flnm,300,6,ifls5a5)
      call match('S77   ',1,6,flnm,300,6,ifls77)
```

```
      call match('S308  ',1,6,flnm,300,6,ifls308)
      call match('S5A4  ',1,6,flnm,300,6,iflbel5)
      call match('S5A3  ',1,6,flnm,300,6,ifll8)
c
      do is = 1,nflpts
        call match (struc_name_meas(is),1,6,flnm,ntstrctr,6,imatch)
        if (imatch .eq. 0) then
          write(*,1000) struc_name_meas(is)
          stop
        endif
        istr_index(is) = imatch
      enddo
 1000 format(2(/),'UNRECOGNIZED MEAS OR BOUNDARY STRUC',2x,A6,2x,'CHECK
     $ MODEL DEFINITION DATA FILE')
c
c
c  read range of dates to output some daily binary output information
c
      if (ip(19) .eq. 1 .or. ip(25) .eq. 1) then
        read(2,*)iyout1,imout1,idout1,iyout2,imout2,idout2
        ijul1 = iymdjl(iyout1,imout1,idout1)
        ijul2 = iymdjl(iyout2,imout2,idout2)
      endif
      DO 40 I = 1,NPOPTS
 40      LP(I) = IP(I) .EQ. 1
c---------------------
c  Land Use Information
C-----------------
      CALL READTK(2,NLU)
C
      do im = 1,12
         READ (2,*) (et_cal_adj_fact(i,im),i=1,nlu)
      enddo
      READ (2,'(30(A7,1X))') (land_use_type(i),i=1,NLU)
      READ (2,*) (KMAX(I),I = 1,NLU)
      READ (2,*) (OWPOND(I),I = 1,NLU)
      READ (2,*) (SRZ(I),I = 1,NLU)
      READ (2,*) (DRZ(I),I = 1,NLU)
      READ (2,*) (OFML(I,1),I = 1,NLU)
      READ (2,*) (OFML(I,2),I = 1,NLU)
      READ (2,*) (OFML_fact(I,1),I=1,NLU)
      READ (2,*) (OFML_fact(I,2),i=1,NLU)
      READ (2,*) (rmin_ofml(I),I=1,NLU)
      READ (2,*) (DETEN_DEF(I),I = 1,NLU)
      READ (2,*) (OFMC(I,1),I=1,NLU)
      READ (2,*) (OFMC(I,2),I=1,NLU)
      READ (2,*) (OFMC(I,3),I=1,NLU)
      READ (2,*) (OFMC(I,4),I=1,NLU)
      READ (2,*) (DETEN_DEFC(I),I = 1,NLU)

c
c  definition parameters for LOK
c
c
      read (2,'(F6.2)') stagelo
      stagelo_res = stagelo
      read (2,'(7(2x,a3),3F6.1)') simcaes,simsles,es_dmnd_acc_freq
```

```
     $            ,opt_bsn_prio_uncond,opt_reg_lok_to_cal_res
     $            ,opt_reg_lok_to_stl_res,opt_prior_use_asr_flex_cal
     $            ,bflo_frac_c43est,frac_cal_dmnd_met_by_del
      read (2,'(2X,A3,2x,2F6.1,2i3)') use_ssm,lok_targ_level,ssmminfrac
     $            ,issm_cutb_opt_bcyp,issm_cutb_opt_brghton
      read (2,'(2X,A3,1x,4F6.2)') pcbruleopt,pcbfrac(1),pcbfrac(2)
     $            ,pcbfrac(3),pcbfrac(4)
      read (2,*) frac_dmnd_met_wt,frac_dmnd_met_wn
      read (2,*) nwgoalsto,(mon_targ_wet(i),iday_targ_wet(i)
     $           ,lok_targ_level_wet1(i),lok_targ_level_wet2(i)
     $           ,i=1,nwgoalsto)
      read (2,*) (rlok_stg_beg_forpmp(i),rlok_stg_end_forpmp(i),i=1,3)
      read (2,*) (forw_pump_cap(i),i=1,3)
      read (2,*) (rmax_tw_eaad(i),i=1,3)
      read (2,*) (rmax_lift(i),i=1,3)
      read (2,*) (rmin_lok_stg_forw_pump(i),i=1,3)
      read (2,*) n_days_week_del_eaa(1),(days_week_del_eaa_ssm(i,1)
     $    ,i=1,n_days_week_del_eaa(1))
      do iw = 1,n_days_week_del_eaa(1)
         call match (days_week_del_eaa_ssm(iw,1),1,3,days_week
     $   ,7,3,imatch)
         idays_wk_del_eaa_ssm_indx(iw,1) = imatch
      enddo
      read (2,*) n_days_week_del_eaa(2),(days_week_del_eaa_ssm(i,2)
     $    ,i=1,n_days_week_del_eaa(2))
      do iw = 1,n_days_week_del_eaa(2)
         call match (days_week_del_eaa_ssm(iw,2),1,3,days_week
     $   ,7,3,imatch)
         idays_wk_del_eaa_ssm_indx(iw,2) = imatch
      enddo
      read (2,*) (frac_ws_bkp_abv_wcaflr(i),i=1,3)
     $              ,frac_bkflw_wpb_via_l8_abv_flr
      read (2,*) (frac_ws_bkp_bel_wcaflr(i),i=1,3)
     $              ,frac_bkflw_wpb_via_l8_bel_flr
      read (2,*) (ws_bkp_cap(i,1),i=1,3),bflo_cap_l8
      read (2,*) (ws_bkp_cap(i,2),i=1,3)
      read (2,*) MAXBPCNTR,iwsbkpwsta
cbw 01-02-01      read (2,'(2X,A3,2x,F6.1)') use_ssm,lok_targ_level
      read (2,'(2x,A4)') demand_level_opt
      read (2,*) ((rlosa_factor(i,j),j=1,2),i=1,8)
      read (2,*) semirr_dmnd_factor,cale_dmnd_factor,sle_dmnd_factor
     $         ,fpl_dmnd_factor
      read (2,*) s235_capac,s235_frac,(diston_capac(i),i=1,2),
     $(diston_frac(i),i=1,2)
      read (2,'(2X,A7,2X,A5,3(2X,A3))') make_up_water_opt
     $         ,makeup_water_restr,bmp_ssm_opt
     $         ,env_ws_cutb_ssm_opt,opt_ws_to_roten_frm_lok
      read (2,'(2X,A4,2x,A3,2i3)') lok_reg_to_wca_mode
     $      ,opt_multi_seas_for_reg_wca,(iopt_coth_wcas(i)
     $      ,i=1,2)
      read (2,*) idbsnopt1,iuse_lok1,iregcals79opt
      read (2,*) idbsnopt2,iuse_lok2,itribres2
      read (2,*) inorth_stor_opt
     $                   ,iprop_res_prior_opt_lokreg
     $                   ,itcns_res_opt
      read (2,*) lok_split_option,fract_mds_res,fract_mds_litzone
     $              ,fract_rain_litzone,capac_struc_to_lokres
```

```
      read (2,*)   caloos_res_area,caloos_res_area_w_asr
     $                  ,stlucie_res_area,rnorth_stor_res_area
     $                  ,tayck_nubsl_res_area,res_seep_factor
      read (2,*) stlucie_res_max_dpth,caloos_res_max_dpth
      read (2,*) isem_flg
      read (2,'(a5,2x,a3)') ssm_env,flow_to_wca_prior
      READ (2,'(15F6.2)') (RSIAPM(I),I=1,12)
      READ (2,'(10I5)') NSTRCA,(irg_to_res_prty(is),IS=1,NSTRCA)
      READ (2,'(10I5)') NSTRCA_REG,(IRGPRTY(IS),IS=1,NSTRCA_REG)
c
c  operational parameters for EAA
c
c
      READ (2,'(3i4)') NEAABSN,neaacnl,NBPSTR
      read (2,'(2f5.2)') solinit,depth_soil_eaa
      read (2,'(5i4)') ISUGBSN,MINYNNR,MAXYNNR
      READ (2,9001)(IXMNNNR(I),I=1,MAXYNNR-MINYNNR+1)
      READ (2,9001)(IXMXNNR(I),I=1,MAXYNNR-MINYNNR+1)
9001  FORMAT(14I3)
      read (2,'(F5.0,2x,A5)') BMPRED,bmp_opt
      read (2,'(10(A5,2x))') hlyenv,rotenenv,nnrctwca3a
     $            ,re_proport_eaa_rnff
      read (2,'(3(A4,2X))') eaa_conv_opt_s7,eaa_conv_opt_s8
      read (2,'(A4,2X,A3)') opt_outflow_from_res_to_wca
     $                  ,sta34_outf_flex_eaar_opt
      read (2,'(A5,2x,A5)') divers_excess_to_res
     $                  ,lrunoff_to_res_when_above_targ
      read (2,'(2(A5,2x))') opt_for_hill_bypass
     $                  ,opt_for_hill_bypass_runoff
      READ (2,9002) (ADJCFF(I),I=1,12)
      DO ib = 1,3
       READ (2,9002) (fracdph_max(i,ib),i=1,12)
       READ (2,9002) (fracdph_min(i,ib),i=1,12)
      ENDDO
      READ (2,'(2i4)') max_days_mean_loktasr,max_days_mean_bkpump
c
c  Holeyland
c
c
      READ (2,9004) (NSINDX(I),I=1,NEAABSN)
      READ (2,9002) (PCFS(I),I=1,NEAABSN)
      READ (2,9002) (PCFWS(I),I=1,NEAABSN)
      READ (2,'(5(A5,1x))') (int_cnl_holey_name(I),I=1,NEAABSN)

      do i = 1,NEAABSN
       call match (int_cnl_holey_name(I),1,5,cnm,nch+1,5,imatch)
       if (imatch .eq. 0) then
          write(*,2029) int_cnl_holey_name(I)
 2029     format(2(/),'UNRECOGNIZED CANAL NAME',2x,A6,2x,'IN HOLEYLAND
     $TO WHICH EXCESS WATER IS DIVERTED. CHECK MODEL DEFINITION DATA F
     $ILE')
          stop
       endif
       int_cnl_holey(i) = imatch
      enddo
      read (2,'(A3,2x,A7,2x,A6)') holey_min_level_opt
     $              ,runoff_to_holeyland,ws_to_holy_opt
```

```
      read (2,'(A5,2x,A4,F6.0)') holy_oper,holy_outflow_opt,hlcnlds
      if (runmode .eq. 'SIMUL') then
        do ib = 1,neaabsn-1
          read (2,'(I5,5(F7.0,F5.2))') nthresholds(ib),(runoff_thres(ib
     $     ,i),pct_daily_runoff(ib,i),i=1,nthresholds(ib))
        enddo
      endif
      READ (2,9002) (PCTRUNF(I),I=1,NEAABSN)
      READ (2,9002) QS7FACT,QS6FACT,QS150FT
      READ (2,9002) (REDBP(I),I=1,2)
c
      DO I = 1,NBPSTR+1
       READ (2,'(5F6.0)') CRRNFF(I),COEFR(I,1),COEFR(I,2)
      ENDDO
c
c  for EAA basins
c
      if (runoff_to_holeyland .eq. 'DIRECT') then
         READ(2,'(5i5)') (ires_pump(i),i=1,neaabsn-1)
      endif
c
c  RES/ASR parameters-EAA basins
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      READ (2,'(7i3)') (ires_opt_eaa(i),i=1,neaabsn+1),ires_opt_hill
      READ (2,'(5i3)') (iasr_opt_eaa(i),i=1,neaabsn)
      READ (2,'(5F5.2)') (frac_runoff_asr(i),i=1,neaabsn)
      READ (2,'(5F5.2)') (frac_avail_asr_to_meet_dmnd(i),i=1,neaabsn)
      READ (2,'(5(A6,1x))') (ieaa_res_asr_name(i),i=1,neaabsn+1)
      do i = 1,neaabsn+1
        read (2,'(i3,2x,5(a6,2x,a5,2x))') no_of_res_reg_frm_lok(i)
     $     ,(ieaa_res_asr_reg_name(i,k),opt_for_reg_rel_to_res(i,k)
     $     ,k = 1,no_of_res_reg_frm_lok(i))
      enddo
      read (2,'(5F6.2)')  (RMAXSTG(I),I=1,NEAABSN)
      read (2,'(5F10.0)') (base_area_eaa_basin(i),i=1,neaabsn-1)
c
c
c
      do i = 1,neaabsn
        read (2,'(i5,2x,10(A6,1x))') n_eaa_str(i),(eaa_str_name(j),j=1,
     $     n_eaa_str(i))
        do j = 1,n_eaa_str(i)
         call match (eaa_str_name(j),1,6,flnm,300,6,imatch)
         if (imatch .eq. 0) then
           write(*,1027) eaa_str_name(j)
 1027      format(2(/),'UNRECOGNIZED EAA STRUC NAME',2x,A6,2X,'CHECK LE
     $CDEF FILE')
         endif
         ieaa_str_indx(i,j) = imatch
        enddo
      enddo
      ifls3   = ieaa_str_indx(1,1)
      ifls8   = ieaa_str_indx(1,2)
      ifls2   = ieaa_str_indx(2,1)
      ifls7   = ieaa_str_indx(2,2)
      ifls6   = ieaa_str_indx(2,3)
```

```
      iflhg5  = ieaa_str_indx(3,1)
      ifls5a  = ieaa_str_indx(3,2)
      ifls150 = ieaa_str_indx(2,4)
      iflagq  = ieaa_str_indx(4,1)
c
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      read (2,'(i5,2x,10(a5,1x))') no_canal_in_l8_basin,
     $  (l8basin_canal_name(i),i=1,no_canal_in_l8_basin)
      do i = 1,no_canal_in_l8_basin
       call match(l8basin_canal_name(i),1,5,cnm,nch+1,5,imatch)
         if (imatch .eq. 0) then
           write(*,1028) l8basin_canal_name(i)
 1028      format(2(/),'UNRECOGNIZED L8 BASIN CANAL NAME',2x,A5,2x,'CHEC
     $K MODEL DEFINITION DATA FILE')
           stop
         endif
       l8basin_canal_indx(i) = imatch
      enddo
      read (2,'(2(A5,2x))') idn_wca_name_l8,ds_int_canal_name_l8
      read (2,'(3F6.1)') (rmin_l8stg_flow_to_mcnl(i),i=1,2)
     $                    ,rmax_wpbcat_for_inflw
      READ(2,'(I1)') opt_l8_prop
      if (opt_l8_prop) then
       il8w= l8basin_canal_indx(1)
       il8 = l8basin_canal_indx(2)
       il8s= l8basin_canal_indx(3)
       if (cnm(l8basin_canal_indx(1)) .ne. canl_names_spec(25)
     $ .or. cnm(l8basin_canal_indx(3)) .ne. canl_names_spec(26))
     $  then
          write(*,1029) cnm(l8basin_canal_indx(1))
     $     ,cnm(l8basin_canal_indx(3))
 1029     format(2(/),'CANAL NAMES',2x,A5,2x,'AND OR',2x,A5,2x,'ARE NOT
     $APPROPRIATE FOR OPTION INPUT FOR CONFIGURATION OF L-8 CANAL. CHECK
     $ THE INPUT FOR CANAL NAMES FOR L-8 BASIN IN model_definition_info.
     $dat FILE OR CHECK OPTION INPUT IN SAME FILE.')
          STOP
       endif
      else
       il8 = l8basin_canal_indx(1)
       il8w = il8
       il8s = il8
       if (cnm(l8basin_canal_indx(1)) .ne. canl_names_spec(24)) THEN
          write(*,1030) l8basin_canal_indx(1)
 1030     format(2(/),'CANAL NAME',2x,A5,2x,'IS NOT APPROPRIATE FOR OPT
     $ION INPUT FOR CONFIGURATION OF L-8 CANAL. CHECK THE INPUT FOR CAN
     $AL NAMES FOR L-8 BASIN IN model_definition_info.dat FILE OR CHECK
     $ OPTION INPUT IN SAME FILE.')
          STOP
       endif
      endif
c
c  for EAA canals
c
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      read (2,*) (RCPFACTEAA_canal(I),I=1,NEAACNL)
```

```
      READ (2,*) (RCPFACTEAA(I),I=1,NEAACNL)
      read (2,*) (rcpfacteaa_res(i),i=1,neaacnl)
      READ (2,*) (frac_lec(i),i=1,neaacnl)
      READ (2,*) (frac_rem_capac(i),i=1,neaacnl)
      READ (2,'(5(A5,2x))') (eaa_conv_canal_names(i),i=1,neaacnl)
      READ (2,'(5(A5,2x))') (idn_wca_name(i),i=1,neaacnl+1)
      READ (2,'(5(A5,2x))') (idn_wca_name_env(i),i=1,neaacnl)
      read (2,'(5(A5,2x))') (ds_convey_c_name(i),i=1,neaacnl+1)
      read (2,'(5(A7,1x))') (opt_pipeline_ws(i),i=1,neaacnl+1)
      ifacteaa = 0
      do i = 1,neaacnl
       if (rcpfacteaa(i) .ne. 1.0 .or. RCPFACTEAA_canal(i)
     $        .ne. 1.0) ifacteaa = 1
      enddo
c
      READ (2,*) (CNLCAP(I),I=1,NEAACNL),CNLCAP(il8bsn-6)
      READ (2,*) (desgn_cap_canl_ws(I),I=1,NEAACNL)
      READ (2,*) (iconv_use_inj(i),i=1,neaacnl)
      do i = 1,neaacnl
        cnlcap(i) = cnlcap(i) * RCPFACTEAA(I)
        desgn_cap_canl_ws(i) = desgn_cap_canl_ws(i) * RCPFACTEAA(I)
      enddo
      do is = 1,2
        READ (2,*) (pct_des(i,is),i=1,NEAACNL)
      enddo
      READ (2,'(5f6.0)') (pmp_int(i),i=1,neaacnl)
      READ (2,'(5f6.0)') (pmp_int_new(i),i=1,neaacnl)
      read (2,*) (iserv_area(i),i=1,neaacnl)
      DO j = 1,neaacnl
        read (2,'(I6,2x,5(A5,2x))') no_targ_loc(j),(targ_area_name(i)
     $    ,i = 1,no_targ_loc(j))
       DO i = 1,no_targ_loc(j)
         call match(targ_area_name(i),1,5,import_area_name,
     $      ntotal_no_import_areas,5,imatch)
         if (imatch .eq. 0) then
            write(*,2030) targ_area_name(i),j
 2030       format(2(/),'UNRECOGNIZED TARGET AREA NAME(RF OPS)',2x,A6,2x
     $ ,'INPUT FOR EAA CANAL NO',2x,I2,2x,'CHECK MODEL DEFINITION DATA F
     $ILE.')
            stop
         endif
         itarg_area(i,j) = imatch
       ENDDO
      ENDDO
c
c  structure capacities,options needed for conveyance calculations
c  and structure flow
c
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      READ (2,*) S3MAX,S2MAX
     $           ,S5A4MAX,RL8AGRFCAP,RMCAGRFCAP,capac_sugh
     $           ,capc_new_pump_to_west_frm_roten,capacs5as
     $           ,facts2mc,frac_sem_cyp,capac_mcnl
     $           ,frac_thru_s8_sta34,frac_c139basin_sta5
     $           ,frac_c139basin_sta6,ic139_rnff_fix_split_opt
      READ (2,*)  capacpsac_ws,capacc10a_bak
```

```
c
c  Water Conservation Areas
c
c
c
      read (2,'(A7)') type_flow_across_ttrail
      read (2,'(A5)') type_target
      read (2,'(A5,2x,A4)') bcnpenv,s332sparrow_opt
      read (2,'(A5)') floor_grid_cell_opt
      read (2,'(A5)') icnl_dn_s31_name
      call match (icnl_dn_s31_name,1,5,cnm,nch+1,5,icnl_dn_s31)
      if (icnl_dn_s31 .eq. 0) then
         write(*,2031) icnl_dn_s31_name
 2031    format(2(/),'UNRECOGNIZED NAME FOR DS CANAL FOR S-31',2x,A5,2x,
     $   'CHECK MODEL DEFINITION DATA FILE.')
         stop
      endif
      READ(2,'(I1)') opt_wca1_prop
      READ(2,'(I1,1x,F5.1)') opt_st_input,allig_alley_res_fact
c
c  Everglades National Park
c
      READ(2,*) main_pres_level_fl_prot_ts
      READ(2,*) use_enp_ws_to_lec
      READ(2,*) opt_flow_to_ts
      READ(2,'(I1)') opt_mrp_to_enp
      READ(2,'(I1)') opt_ft_input
c
c  Service Areas - Water Supply
c
c
c
C-----------------
      READ (2,*)    NCNL_WS_MIN,(NCL_WS_NAME(I),NDS_CNL_WS_NAME(I)
     $                ,I=1,NCNL_WS_MIN)
      DO i = 1,NCNL_WS_MIN
        call match(NCL_WS_NAME(I),1,5,cnm,nch+1,5,imatch)
        ncl_no(i) = imatch
        call match(NDS_CNL_WS_NAME(I),1,5,cnm,nch+1,5,imatch)
        nds_cnl_no(i) = imatch
      ENDDO
c
      read (2,*) no_flwth_cnls,(cnl_flwth_name(i),i = 1
     $              ,no_flwth_cnls)
      read (2,*) no_fc_flwth_cnls,(cnl_fc_flwth_name(i),i = 1
     $              ,no_fc_flwth_cnls)
      do i = 1,no_flwth_cnls
       call match(cnl_flwth_name(i),1,5,cnm,nch+1,5,imatch)
       if (imatch .eq. 0) then
          write(*,2032) cnl_flwth_name(i)
 2032     format(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,',INPUT AS FLO
     $W THROUGH FOR WATER SUPPLY PURPOSES. CHECK MODEL DEFINITION DATA F
     $ILE.')
          stop
       endif
       icnl_flwth(i) = imatch
c      print *,i,no_flwth_cnls,cnl_flwth_name(i),icnl_flwth(i)
```

```
c    $   ,'  CFLWTH'
      enddo
      do i = 1,no_fc_flwth_cnls
        call match(cnl_fc_flwth_name(i),1,5,cnm,nch+1,5,imatch)
       if (imatch .eq. 0) then
          write(*,2033) cnl_flwth_name(i)
 2033     format(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,',INPUT AS FLO
     $W THROUGH FOR FLOOD CONTROL PURPOSES. CHECK MODEL DEFINITION DATA
     $FILE.')
          stop
       endif
        icnl_fc_flwth(i) = imatch
      enddo
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      READ (2,*) NSVAREA
      DO I = 1,NSVAREA
        READ(2,'(i5,2x,10(A5,1x))') nsacnl(i),(sa_canal_name(j)
     $         ,j=1,nsacnl(i))
        DO J = 1,nsacnl(i)
         call match(sa_canal_name(j),1,5,cnm,nch+1,5,imatch)
         if (imatch .eq. 0) then
           write(*,2034) sa_canal_name(j),I
 2034      format(2(/),'UNRECOGNIZED RECIPIENT CANAL NAME',2x,A5,2x,',FO
     $R WATER SUPPLY DISCHARGES FROM APPROP WCA TO SERVICA AREA',1x,i2,
     $2x,'CHECK MODEL_DEFINITION_DATA FILE')
           stop
         endif
         isa_canal_indx(i,j) = imatch
        ENDDO
      ENDDO
      DO 9983 I = 1,NSVAREA
       READ (2,'(I5,2x,20(A5,1x))') NSTART(I),(CISTC(J),J = 1,NSTART(I))
       DO J = 1,NSTART(I)
        call match(CISTC(J),1,5,cnm,nch+1,5,imatch)
        if (imatch .eq. 0) then
         write(*,1007) CISTC(J)
         stop
        endif
        ISTC(J,I) = imatch
       ENDDO
 1007  format(2(/),'UNRECOGNIZED INITIAL CANAL NAME FOR WSNEEDS',2x,A5,
     $ 2x,'CHECK MODEL_DEFINITION_DATA FILE')
 9983 CONTINUE
      read (2,'(F6.1,2x,A5)') rmin_stg_l30_ws,loxenv
c
c   input data for Water Reuse
c
      read (2,*) no_of_reuse_plnts
       do i = 1,no_of_reuse_plnts
        read (2,*) (avg_daily_reuse_vol(i,j),j=1,12),iopt_rec_reuse(i)
        if (iopt_rec_reuse(i) .eq. 0) then
         read (2,'(i5,5(2x,A5))')no_canals_reuse(i)
     $         ,(canal_reuse_names(k),k=1,no_canals_reuse(i))
         do k=1,no_canals_reuse(i)
           call match(canal_reuse_names(k),1,5,cnm,nch+1,5,imatch)
           if (imatch .eq. 0) then
             write(*,1010) canal_reuse_names(k)
```

```
 1010  format(2(/),'UNRECOGNIZED  CANAL NAME FOR REUSE',2x,A5,2x,
     $ 'CHECK MODEL_DEFINITION_DATA FILE')
             stop
           endif
           icanal_reuse_indx(i,k) = imatch
         enddo
        else
         read (2,'(5(3i5,2x,a6,2x,F4.1))') no_grid_cells_reuse(k)
     $       ,(icol_reuse(k),irow_reuse(k),resname_reuse(i,k)
     $       ,rmax_stage_reuse(i,k),k=1,no_grid_cells_reuse(i))
         do k=1,no_grid_cells_reuse(i)
           node_reuse(i,k) = icol_reuse(k) - minx(irow_reuse(k))
     $          + 1 + isum(irow_reuse(k))
         enddo
        endif
       enddo
c
c  special input for execution of locwslwdd subroutine
c  determining demands within Lake Worth Drainage District(LWDD)
c
      READ (2,'(6F6.0)') CAPACCS2,CAPACCS4,CAPACCS9,CAPACCS12,CAPACCS17W
     $,                    CAPACCS17E
      READ (2,'(I5,2x,10(A6,1x))') nstr_lwdd,(str_lwdd_name(i)
     $     ,i=1,nstr_lwdd)
      READ (2,'(I5,2x,10(A5,1X))') n_up_canals_lwdd
     $   ,(iup_canals_lwdd_name(i),i = 1,n_up_canals_lwdd)
      READ (2,'(I5,2x,10(A5,1X))') n_canals_lwdd,(canal_lwdd_name(i)
     $         ,i = 1,n_canals_lwdd)
      READ (2,'(2(A5,1x))') idowns_wca_name_lwdd,int_cnl_name_for_lwdd
      do i = 1,nstr_lwdd
        call match(str_lwdd_name(i),1,6,flnm,300,6,imatch)
        if (imatch .eq. 0) then
          write(*,123) str_lwdd_name(i)
 123      format(2(/),'UNRECOGNIZED STRUC NAME FOR LWDD',2x,A6,2x,
     $    'CHECK MODEL_DEFINITION_DATA FILE')
          stop
        endif
        istr_indx_lwdd(i) = imatch
      enddo
      do i = 1,n_up_canals_lwdd
       call match(iup_canals_lwdd_name(i),1,5,cnm,nch+1,5,imatch)
       if (imatch .eq. 0) then
         write(*,124) iup_canals_lwdd_name(i)
 124     format(2(/),'UNRECOGNIZED UPSTREAM CANAL NAME FOR LWDD',2x,A5
     $   ,2x,'CHECK MODEL_DEFINITION_DATA FILE')
         stop
       endif
       iup_canal_indx_lwdd(i) = imatch
      enddo
      do i = 1,n_canals_lwdd
       call match(canal_lwdd_name(i),1,5,cnm,nch+1,5,imatch)
       if (imatch .eq. 0) then
        write(*,126) canal_lwdd_name(i)
 126    format(2(/),'NRECOGNIZED CANAL NAME IN LWDD',2x,A5,2x,
     $  'CHECK MODEL_DEFINITION_DATA FILE')
        stop
       endif
```

```
       icanal_indx_lwdd(i) = imatch
      enddo
      IE1 = icanal_indx_lwdd(1)
      IE2 = icanal_indx_lwdd(2)
      IE3 = icanal_indx_lwdd(3)
      IES = icanal_indx_lwdd(4)
      IESE = icanal_indx_lwdd(5)
      IC51 = iup_canal_indx_lwdd(1)
      LHL  = iup_canal_indx_lwdd(2)
c
      IFLCS2   = istr_indx_lwdd(2)
      IFLCS9   = istr_indx_lwdd(3)
      IFLCS12  = istr_indx_lwdd(4)
      IFLCS17W = istr_indx_lwdd(5)
      IFLCS17E = istr_indx_lwdd(6)
      IFLCS4   = istr_indx_lwdd(7)
c
c     Input information for routing of 298 District runoff to STAs
c
      READ (2,*) no_of_298d_basins_t_stas
      DO i = 1,no_of_298d_basins_t_stas
         READ(2,*) frac_298r_s(i),struc_cap_298d_s(i)
     $     ,name_eaa_cond_298_s(i),rmax_cap_298d_to_lok(i)
         call match (name_eaa_cond_298_s(i),1,5,
     $    eaa_conv_canal_names,neaacnl,5,imatch_cn)
         if (imatch_cn .eq. 0) then
            write (*,2035) name_eaa_cond_298_s(i)
 2035       format(2(/),'UNRECOGNIZED EAA CANAL NAME',2x,
     $      A5,2x,'FOR 298 DISTRICT OUTFLOW. CHECK model_definition_info
     $.dat FILE.')
            stop
         endif
         idest_eaacnl_298d(i) = imatch_cn
      ENDDO
c
      READ (2,*) iopt_ws_lec_to_sta_wca_abv_flr
     $  ,iroute_l8rnf_to_sta_opt
C
      READ (2,'(I5,15F6.0)') NBSNTSTA,(PCTWMA(I),I=1,NBSNTSTA)
      IF (NBSNTSTA .GT. 0) THEN
       READ (2,'(20(A6,1x))') (name_res_for_inflow(i),I=1,NBSNTSTA)
       READ (2,9004) (ISTAOPT(I),I=1,NBSNTSTA)
      ENDIF
 9002  FORMAT (30F6.0)
 9004  FORMAT(50I6)
       IF (ISTAOPT(10) .eq. 0) ISUGBSN = IS8BSN
c
c  input locations (col,row) of monitoring points to be output
c   to tape76
c
c
      read (2,*) nmtr
      DO I = 1,NMTR
        READ (2,'(A5,I5,20(1X,2I3))') PLTNM(I,1),n_cells(I),(IX(k),IY(k)
     $   ,k=1,n_cells(I))
        do k = 1,n_cells(I)
          MTRP(I,k) = IX(k) - MINX(IY(k)) + 1 + ISUM(IY(k))
```

```
        enddo
      enddo
      CLOSE(2)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  WRITE OUT FLOW-TO POINTS TO BE SURE THEY GOT INTO THE NODECN ARRAY
C-----------------
      NPTS = 0
      DO  I = 1,NCH
      DO  J = 1,NODCR(I)
      NPTS = NPTS + 1
      DO 8 KCN = 1,MAXY
       IF (NODECN(NPTS)/10 .LE. ISUM(KCN)) GO TO 9
   8  CONTINUE
   9   YCN(I,J) = KCN - 1
       XCN(I,J) = NODECN(NPTS)/10 - 1 + MINX(YCN(I,J)) - ISUM(YCN(I,J))
       ICL(I,J) = MOD ( NODECN(NPTS) , 10)
      ENDDO
      ENDDO
C-----------------
C
C   READ TABLE FOR EAA CANAL WATER SURFACE PROFILE DATA FROM UNIT 13
C-----------------
C
      read(13,'(i5)')nreach
      do k = 1,nreach
         read(13,'(3i5)')ireach,ntws(ireach),n_eaaflows(ireach)
         nj = ntws(ireach)
         nq = n_eaaflows(ireach)
         do j = 1, nj
            read(13,'(f10.0)')dsstage(j,ireach)
            do i = 1, nq
               read(13,'(2f10.0)') eaaflow(i,j,ireach),
     &                             usstage(i,j,ireach)
            enddo
         enddo
      enddo
      close(13)
C-----------------
C
C READ MAXIMUM GATE OPENING DATA FOR EAA SPILLWAYS
C (S354,S351,S352)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
      read(15,'(i5)')nstr_go
      do k = 1,nstr_go
         read(15,'(3i5)')istr_go,ntails(istr_go),n_strgos(istr_go)
         nj = ntails(istr_go)
         ngo = n_strgos(istr_go)
         do j = 1, nj
            read(15,'(f10.0)')dsstg(j,istr_go)
            do i = 1, ngo
               read(15,'(2f10.0)') strgo(i,j,istr_go),
     &                             usstg(i,j,istr_go)
            enddo
         enddo
      enddo
```

```
      close(15)
C
C-----------------
      RETURN
      END
```

gen_model_run_def_param.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: gen_model_run_def_param.F,v 1.13 2003/11/17 19:41:17 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/gen_model_run_def_param.F,v $
C   $Name: rel-5-5 $
C
C
C   subroutine to retrieve model run definition data required by model
C   file opened:  gen_model_def_param.dat
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE gen_model_run_def_param (xspc, yspc, ibm_initc, npopts,
     & maxy_nsm, minx_nsm, maxx_nsm, isum_nsm, isyr, ip, isyretbin
     &,iseas0)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'add_misc.inc'
      INCLUDE 'climvar.inc'

      INTEGER ip(60), isum_nsm(max_n_rows_incl_lok),
     & maxx_nsm(max_n_rows_incl_lok), maxy_nsm,
     & minx_nsm(max_n_rows_incl_lok)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C      set num_use_type = 0.  It is reset to the number of irrigation
```

```
C      use types in SUBROUTINE lec_et_initial_READ IF use_lec_et.ne.0

      num_use_type = 0

      READ (112,'(4i5)') ifyr,ibm,iENDyr,ilastm
      nyear = iENDyr - ifyr + 1
      iem = (iENDyr - ifyr) * 12 + ilastm
      READ (112,'(3I5)') maxy,maxy_nsm
      READ (112,'(2F5.0)') xspc,yspc
      READ (112,'(F5.1)') DT
      READ (112,'(I4)') ntstep_ov
      READ (112,'(I4)') max_nca_ov
      READ (112,'(A5)') runmode
      READ (112,'(A7)') typerun
      READ (112,'(9I4)') npopts, initialize_annually_opt, ibm_initc,
     & ifreq_init
      READ (112,'(2I5,2I3)') isyr,isyretbin,ism
      READ(112,*) use_lec_et, use_trigger,wca_import_variation
c
c   define initial season (wet or dry) based on initial month
c
      if (ibm .gt. 5 .and. ibm .le. 10) then
         iseas0 = 2
      else
         iseas0 = 1
      endif
C  input min and max column for each row of grid system

      READ (112,'(14i3)') (minx(i),i=1,maxy)
      READ (112,'(14i3)') (maxx(i),i=1,maxy)
      READ (112,'(14i3)') (minx_nsm(i),i=1,maxy_nsm)
      READ (112,'(14i3)') (maxx_nsm(i),i=1,maxy_nsm)
      READ (112,'(14i3)') (MXOV(I),I=1,MAXY)
      DO i = 1,npopts
         READ (112,'(I2)') ip(i)
      END DO
c
      IF       ( IP(1) .eq. 1  ) THEN
         OPEN(55,FILE = 'restart_output')
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      ISUM (1)    = 0
      MAXXT       = 0
      DX = 5280. * XSPC
      DY = 5280. * YSPC
C-----------------
C  DETERMINE MAXXT & CALCULATE ISUM ARRAY
C-----------------
      DO 10 I = 1,MAXY
         IF (MAXX(I) .GT. MAXXT ) MAXXT = MAXX(I)
         ISUM(I+1) = ISUM(I) + MAXX(I) - MINX(I) + 1
   10 CONTINUE
c
      ISUM_NSM(1) = 0
c
c  DETERMINE ISUM ARRAY FOR NSM GRID RAINFALL
```

```
c   FOR MAPPING NSM RAIN INTO SFWMM GRID NETWORK
c
      DO I = 1,MAXY_NSM
         ISUM_NSM(I+1) = ISUM_NSM(I) + MAXX_NSM(I) - MINX_NSM(I) + 1
      END DO
c
      RETURN
      END
```

grid_cell_based_data.F

```
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: grid_cell_based_data.F,v 1.21 2005/06/10 13:29:06 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/grid_cell_based_data.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO RETRIEVE GRID CELL BASED PHYSICAL DATA
C                  REQUIRED BY MODEL
C   FILES OPENED:
C    static_grid_values.dat   GRID CELL BASED STATIC DATA FILE
C
C   INPUT VARIABLES USED:
C    ELLS()    TOPOGRAPHY INFORMATION FOR EACH NODE POINT (FT NGVD)
C    S()       STORAGE COEFFICIENT (DIMENSIONLESS)
C    LUTYP()   LAND USE INDICATORS FOR EACH NODE
C    IBSN()    SURFACE WATER BASIN INDICATORS
C    H()       (INITIAL) WATER TABLE POSITION (FT NGVD)
C    POND()    (INITIAL) SURFACE WATER DEPTHS  (FT ABOVE LAND SURFACE)
C    AQDEP()   AQUIFER THICKNESS FROM BOTTOM OF FORMATION TO NGVD (FT)
C    TKX()     AQUIFER PERMEABILITY VALUES AT EACH NODE (FT/DAY*10^-4)
C    IETZON()        ET BASIN INDICATORS FOR EACH NODE
C    SINF()          INFILTRATION RATE (FT/DAY)
C    LANDSCAPE()     LANDSCAPE IRRIGATED AREAS (ACRES)
C    NURSERY()       NURSERY IRRIGATED AREAS (ACRES)
C    GOLF_COURSE()   GOLF COURSE IRRIGATED AREAS (ACRES)
C    AG_LOW_VOLUME() AGR_LOW_VOLUME IRRIGATED AREAS (ACRES)
C    AG_OVERHEAD()   AGR_OVERHEAD IRRIGATED AREAS (ACRES)
C    AG_OTHER()      AGR_OTHER IRRIGATED AREAS (ACRES)
C    ERZD()          EFFECTIVE ROOT ZONE (FT)
C    SMMAX()         MAXIMUM SOIL MOISTURE IN THE UNSATURATED ZONE (AC-FT)
C    FLIRPWS()       FRACTION OF LANDSCAPE IRRIGATION RECEIVING WATER
C                    FROM PUBLIC WATER SUPPLY WELLS
C    FGIRTWW()       FRACTION OF GOLF COURSE IRRIGATION FROM TREATED
C                    WASTEWATER
C    FLIRTWW()       FRACTION OF LANDSCAPE IRRIGATION FROM TREATED
C                    WASTEWATER
```

```
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE grid_cell_based_data (sumrain, tareaeaa, solmdph,
     & pond0, solmx0, hinit, rinit_wca_stage, base_area_eaa_basin,
     & ibasin_no_wca, grid_area_res, depth_soil_eaa,
     & initialize_annually_opt, solinit, fracdph_min, fracdph_max,
     & n_add_lev_sys, total_afact,no_nonscane_irr_cells_l8,ofml2,
     & rmin_ofml2,total_area_298d_grid)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'smallres.inc'

      DIMENSION base_area_eaa_basin(max_n_eaa_condt),fracdph_max(12,3),
     & fracdph_max_daily(366), fracdph_min(12,3),fracdph_min_daily(366),
     & grid_area_res(max_n_res), hinit(max_ncells),
     & ibasin_no_wca(max_n_wcas), pond0(max_ncells),
     & rinit_wca_stage(max_n_wcas), solmdph(max_ncells),
     & solmx0(max_ncells), sumrain(max_ncells), total_afact(max_ncells),
     & ofml2(max_ncells),rmin_ofml2(max_ncells),
     & ixerr_irrig(max_ncells),iyerr_irrig(max_ncells),
     & fracdph_max_b(12),fracdph_min_b(12)

      CHARACTER bb*120

      REAL midmon(12)

      DATA bb/'
     &                              '/
      DATA midmon /15.0, 46.0, 74.0, 105.0, 135.0, 166.0, 196.0, 227.0,
     & 258.0, 288.0, 319.0, 349.0/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  Initialize reservoir variables

      DO ia = 1,ntotres+n_add_lev_sys
         grid_area_res(ia) = 0.0
      END DO
      no_nonscane_irr_cells_l8 = 0

      NDATA = 11

C  READ LAND SURFACE ELEVATION DATA   (ELLS ARRAY)
```

```
      IF (LP(5)) WRITE (75,10)

      DO 40 I = MAXY,1,-1
         NDPTS = ISUM(I + 1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF(NPTS .LT. 0) NPTS = 0
         NREC = (NDPTS - 1)/NDATA + 1
         DO 20 J = 1,NREC
            READ(60,710) JY,IX1,IX2,( ELLS(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY) + 1, ISUM(JY)+IX2-MINX(JY) + 1)

   20    CONTINUE
         IF (LP(5)) THEN
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            WRITE(75,30) B(1:LL),( ELLS(NODE), NODE = ISUM(I)+1+NPTS,
     $      ISUM(I+1) )
         ENDIF
   40 CONTINUE
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 50 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 50
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,30) BB(1:LL),( ELLS(NODE), NODE = ISUM(I) + 1, ISUM
     $      (I)+NPTS )
   50    CONTINUE
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
C-----------------
C  READ STORAGE COEFFICIENT  (DIMENSIONLESS)
C-----------------
      IF (LP(5)) WRITE (75,60)
C
      DO I = MAXY,1,-1
         NDPTS = ISUM(I + 1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         NREC  = (NDPTS - 1)/NDATA + 1
         DO J = 1,NREC
            READ(60,710) JY,IX1,IX2,( S(NODE), NODE = ISUM(JY)+IX1-MINX
     $      (JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

         END DO
         IF (LP(5)) THEN
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            WRITE (75,30) B(1:LL),( S(NODE), NODE = ISUM(I)+1+NPTS, ISUM
     $      (I+1) )
         ENDIF
      END DO
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 70 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 70
            LL = (MINX(I) - 1) * 5 + 1
```

```
            NPTS = 15 - MINX(I)
            WRITE(75,30)BB(1:LL),( S(NODE), NODE = ISUM(I) + 1, ISUM(I)
     $      +NPTS )
   70    CONTINUE
      ENDIF
C
C-----------------
      NDATA = 27
C
C  READ LAND USE INDICATORS   (LUTYP ARRAY)
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (LP(5)) WRITE (75,80)
C
      DO 110 I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         NREC  = (NDPTS - 1)/NDATA + 1
         DO 90 J=1,NREC
            READ(60,720) JY,IX1,IX2,( LUTYP(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY) + 1, ISUM(JY)+IX2-MINX(JY) + 1)

   90    CONTINUE
  110 CONTINUE
c
      DO IA = 1,ntotres+n_add_lev_sys
        IF (IA .gt. ntotres .OR. sfactor(ia) .ge. sfactmin
     $   .OR. ires_small_sim(IA) .eq. 'NO ') THEN
          IF (nnodes(IA) .GT. 0) THEN
             DO N = 1,nnodes(IA)
                LUTYP(node_reserv(ia,n)) = lutyp_res_loc(IA,N)
             ENDDO
          ENDIF
        ENDIF
      ENDDO
      DO I = MAXY,42,-1
        print *,I,(LUTYP(NODE),NODE=ISUM(I)+1,ISUM(I+1))
      ENDDO
C
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 120 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 120
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,700) BB(1:LL),( LUTYP(NODE), NODE = ISUM(I)+1, ISUM
     $      (I)+NPTS )
  120    CONTINUE
      ENDIF
      NDATA = 27
C
C-----------------
C  READ BASIN IDENTIFIERS   (IBSN ARRAY)
C-----------------
      DO 130 I = 1,NEAABSN
         NAGNODE(I) = 0.
```

```
  130 NNODES_WETL(I) = 0
      IF (LP(5)) THEN
         WRITE (75,140)
      ENDIF
C
      NAGNODEL8 = 0.
      NAGNODESG = 0.
      NNODESSG_WETL = 0.
      TAREAEAA   = 0.0
      TOTMIAMI   = 0.0
      total_area_298d_grid = 0.0
      TOTNNRHIL  = 0.0
      TOTWPB     = 0.0
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      DO 170 I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF(NPTS .LT. 0) NPTS = 0
         NREC  = (NDPTS-1)/NDATA + 1
         DO 150 J = 1,NREC
            READ(60,720) JY,IX1,IX2,( IBSN(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)
  150    CONTINUE
c
  170 CONTINUE
      DO IA = 1,ntotres+n_add_lev_sys
         DO N = 1,nnodes(IA)
            IBSN(node_reserv(ia,n)) = ibsn_no_res(IA)
            grid_area_res(ia) = grid_area_res(ia) + gdar/43560.
         ENDDO
      ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c  compute ratio of actual reservoir area to reservoir area
c  according to grid system
      DO ia = 1,ntotres
         sfactor(ia) = res_area(ia) / grid_area_res(ia)
         adjust_factr_area(ia) = (1.0 - sfactor(ia))/sfactor(ia)
      END DO
C
c  compute total fraction of reservoir area for each grid cell
c
      DO node = 1,isum(maxy+1)
        total_afact(node) = 0.0
        total_small_afact(node) = 0.0
        DO k = 1,noresincell(node)
           ia = ires_index_in_cell(node,k)
           IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &     (ia) .eq. 'NO ') THEN
              total_afact(node) = total_afact(node) +
     &        sfactor(ia)
           ELSE
              total_small_afact(node) = total_small_afact(node)
     &        + sfactor(ia)
           ENDIF
           IF (total_small_afact(node) .gt. 0.000001 .and.
```

```
     &      total_afact(node) .gt. 0.000001) then
              write(*,3) resname(ia),node
  3           format(/,'HAVE A LARGE AND A SMALL RESERVOIR IN SAME CELL.
     & NEED TO MOVE ONE OF THEM TO ANOTHER CELL LOCATION',2x,'ONE OF THE
     & RESERVOIRS NAME IS',2X,A6,2X,'CELL NUMBER IS',2x,I4)
              stop
            ENDIF
            IF (total_small_afact(node) .GT. sfactmin) THEN
              write (*,4) node,resname(ia)
  4           format(/,'TOTAL AREA COVERAGE OF SMALL RESERVOIRS IN CELL
     &NO',1x, I5,2x,'EXCEEDS RECOMMENDED FRACTION. LAST RESERVOIR INPUT
     &FOR CELL IS',2x,A6)
              stop
            ENDIF
          ENDDO
        ENDDO
c
        DO I = MAXY,1,-1
           IF (I .GT.41)
     $     print *,I,(ibsn(node),node=isum(i)+1,isum(i+1))
           DO NODE = ISUM(I)+1,ISUM(I+1)
              IF (IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 10 ) THEN
                 IF (land_use_type(LUTYP(NODE)) .NE. 'WETLAND') THEN
                    NAGNODE(IBSN(NODE)-6) = NAGNODE(IBSN(NODE)-6) + 1.
                 ELSE
                    NNODES_WETL(IBSN(NODE)-6) = NNODES_WETL(IBSN(NODE)-6)
     $              + 1
                 ENDIF
              ENDIF
              IF (IBSN(NODE) .NE. 0) THEN
                 NNDREC(IBSN(NODE)) = NNDREC(IBSN(NODE)) + (1.0 -
     $           total_small_afact(node))
              ENDIF
              IF (IBSN(NODE) .EQ. 7 .OR. IBSN(NODE) .EQ. 42) THEN
                 TOTMIAMI = TOTMIAMI + GDAR*(1.0-total_small_afact(node))
     $           / 43560.0
              ENDIF
              IF (IBSN(NODE) .EQ. 10) THEN
                 total_area_298d_grid = total_area_298d_grid + GDAR *
     $           (1.0-total_small_afact(node)) / 43560.0
              ENDIF
              IF (IBSN(NODE) .EQ. 8) THEN
                 TOTNNRHIL = TOTNNRHIL + GDAR*(1.0-total_small_afact
     $           (NODE)) / 43560.0
              ENDIF
              IF (IBSN(NODE) .EQ. 9) THEN
                 TOTWPB = TOTWPB + GDAR *(1.0-total_small_afact(node))
     $           / 43560.0
              ENDIF
              IF ((IBSN(NODE) .GE.  7 .AND. IBSN(NODE) .LE. 10) .OR. IBSN
     $        (NODE) .EQ. 42) THEN
                 TAREAEAA = TAREAEAA + GDAR *(1.0 - total_small_afact
     $           (node)) / 43560.0
              ENDIF
              IF (IBSN(NODE) .EQ. 40) THEN
                 NAGNODEL8 = NAGNODEL8 + 1.
              ENDIF
```

```
            IF (IBSN(NODE) .EQ. 42) THEN
               IF (land_use_type(LUTYP(NODE)) .NE. 'WETLAND') THEN
                  NAGNODESG = NAGNODESG + 1.
               ELSE
                  NNODESSG_WETL = NNODESSG_WETL + 1
               ENDIF
            ENDIF
         ENDDO
      ENDDO
        print *,TOTMIAMI,TOTNNRHIL,TOTWPB,TAREAEAA,NAGNODESG
     &  ,'  EAAR  ',NAGNODE(1),NAGNODE(2),NAGNODE(3),NAGNODE(4)
c
c  compute ratio of irrigated area for each major EAA basin
c  for simulation to that with no STAs or prop Reservoirs.
c  This is needed to compute proper runoff reduction for
c  simulation (alternative).
c
      DO ib = 1,neaabsn-1
         IF (ib .EQ. 1) THEN
            frac_area_eaa_basin(ib) = totmiami / base_area_eaa_basin(ib)

         ELSE IF (ib .EQ. 2) THEN
            frac_area_eaa_basin(ib) = totnnrhil / base_area_eaa_basin
     $      (ib)
         ELSE
            frac_area_eaa_basin(ib) = totwpb / base_area_eaa_basin(ib)

         ENDif
      END DO
      frac_area_eaa_bmp = (totmiami+totnnrhil+totwpb) /
     $(base_area_eaa_basin(1)+base_area_eaa_basin(2) +
     $base_area_eaa_basin(3))
c
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 180 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 180
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,100) BB(1:LL),( IBSN(NODE), NODE = ISUM(I)+1, ISUM
     $      (I)+NPTS )
  180    CONTINUE
      ENDIF
C
C-----------------
      NDATA = 11
C
C  READ INITIAL GROUND WATER ELEVATIONS   (FT NGVD)
C-----------------
      IF (LP(5)) WRITE (75,190)
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (LP(1)) THEN
         DO 210 I = MAXY,1,-1
            LL = (MINX(I) - 1) * 5 + 1
            READ(55,280) B(1:LL),(H(NODE),NODE=ISUM(I)+1,ISUM(I+1))
            NDPTS = ISUM(I+1) - ISUM(I)
```

```
            NREC = (NDPTS - 1)/ NDATA + 1
            DO 200 J = 1,NREC
  200       READ(60,*)
  210    CONTINUE
         DO 220 I = MAXY,1,-1
            LL = (MINX(I) - 1) * 5 + 1
            READ(55,280) B(1:LL),(POND(NODE) ,NODE=ISUM(I)+1,ISUM(I+1))
            DO NODE=ISUM(I)+1,ISUM(I+1)
              DO ir = 1,noresincell(node)
                 stage_res(node,ir) = h(node) + pond(node)
                 pond_in_res(node,ir) = amax1(stage_res(node,ir)
     $           - ells_in_res(node,ir),0.0)
              ENDDO
            ENDDO

  220    CONTINUE
      ELSE
         DO 240 I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC  = (NDPTS-1)/NDATA + 1
            DO 230 J = 1,NREC
               READ(60,710) JY,IX1,IX2,( H(NODE), NODE = ISUM(JY)+IX1
     $         -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

               DO NODE = ISUM(JY)+IX1-MINX(JY)+1,ISUM(JY)+IX2-MINX(JY)+1
                  IF (H(NODE) .GT. ELLS(NODE)) THEN
                     POND(NODE) = H(NODE) - ELLS(NODE)
                     H(NODE)    = ELLS(NODE)
                  ELSE
                     POND(NODE) = 0.0
                  ENDIF
                  DO ir = 1,noresincell(node)
                     iresnum = ires_index_in_cell(node,ir)
                     IF(rinit_res_stg(iresnum) .lt. -900.) THEN
                       stage_res(node,ir) = H(NODE) + POND(NODE)
                       pond_in_res(node,ir) = amax1(stage_res(node,ir)
     $                 - ells_in_res(node,ir),0.0)
                     ELSE
                       stage_res(node,ir) = rinit_res_stg(iresnum)
                       pond_in_res(node,ir) = amax1(stage_res(node,ir)
     $                 - ells_in_res(node,ir),0.0)
                     ENDIF
                  ENDDO
               END DO
C
  230       CONTINUE
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,30) B(1:LL),( H(NODE), NODE = ISUM(I)+1
     $      +NPTS, ISUM(I+1) )
  240    CONTINUE
      ENDIF
C
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 250 I = MAXY,1,-1
```

```
            IF (MINX(I) .GE. 15) GO TO 250
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,30) BB(1:LL),( H(NODE), NODE = ISUM(I)+1, ISUM(I)
     $      +NPTS )
  250    CONTINUE
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C  SET EAA OPERATIONAL PARAMETERS AND INITIAL CONSERVATION AREA STAGES
C
c     DO jday = 1,365
c        fracdph_max_daily(jday) = ainterp(REAL(JDAY) ,MIDMON,
c    $   fracdph_max,12,14.0,16.0,1)
c        fracdph_min_daily(jday) = ainterp(REAL(JDAY) ,MIDMON,
c    $   fracdph_min,12,14.0,16.0,1)
c     END DO
c     fracdph_max_daily(366) = (fracdph_max_daily(365) +
c    $fracdph_max_daily(1)) / 2.0
      DO 270 I = MAXY,1,-1
         DO 260 J = ISUM(I) + 1,ISUM(I + 1)
            IF ((IBSN(J) .GE. 7 .AND. IBSN(J) .LE. 10 .AND.
     $      land_use_type(LUTYP(J)) .NE. 'WETLAND') .OR. IBSN(J)
     $      .EQ .40 .OR. IBSN(J) .EQ. 42) THEN
               SOLMDPH(J) = depth_soil_eaa * s(j)
               IAR0 = IBSN(J) - 6
               IAR1 = IAR0
               IF (IBSN(J).EQ.40 .OR. IBSN(J).EQ.
     $          10) THEN
                 IAR1 = 3
               ELSEIF (IBSN(J).EQ.42) THEN
                 IAR1 = 1
               ENDIF
               DO imm = 1,12
                 fracdph_max_b (imm) = fracdph_max(imm,iar1)
                 fracdph_min_b (imm) = fracdph_min(imm,iar1)
               ENDDO
               DO jday = 1,365
                 fracdph_max_daily(jday) = ainterp(REAL(JDAY) ,MIDMON,
     $           fracdph_max_b,12,14.0,16.0,1)
                 fracdph_min_daily(jday) = ainterp(REAL(JDAY) ,MIDMON,
     $           fracdph_min_b,12,14.0,16.0,1)
               END DO
               fracdph_max_daily(366) = (fracdph_max_daily(365) +
     $         fracdph_max_daily(1)) / 2.0
               DO julday = 1,366
                  solcrnf(j,julday) = fracdph_max_daily(julday)*solmdph
     $            (j)
                  solcrt (j,julday) = fracdph_min_daily(julday)*solmdph
     $            (j)
               END DO
            ENDIF
c
            IF (.NOT. LP(1)) THEN
c              IF (runmode .EQ. 'CALIB') THEN
c                 DO ISA = 1,NWCAS
c                    IF (IBSN(J) .EQ. ibasin_no_wca(isa)) THEN
```

```
c                       POND(J) = rinit_wca_stage(isa) - ells(j)
c                       IF (POND(J) .LE. 0.) THEN
c                          h(j) = rinit_wca_stage(isa)
c                       ENDif
c                    ENDIF
c                 END DO
c              ENDIF
               IF ((IBSN(J) .GE. 7 .AND. IBSN(J) .LE. 10 .AND.
     $         land_use_type(LUTYP(J)) .NE. 'WETLAND') .OR. IBSN(J).EQ.
     $         40 .OR. IBSN(J) .EQ. 42) THEN
                 POND(J) = 0.0
                 H(J) = ELLS(J) - SOLMDPH(J) / S(J)
               ENDIF
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (POND(J) .LT. 0.) POND(J) = 0.
            IF (POND(J) .GT. 0.)  H (J)  = ELLS(J)
  260    CONTINUE
         IF (.NOT. LP(5)) GO TO 270
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         LL = (MINX(I) + NPTS - 15) * 5 + 1
         WRITE(75,30) B(1:LL),( POND(NODE), NODE = ISUM(I)+1+NPTS, ISUM
     $   (I+1) )
  270 CONTINUE
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 290 I = MAXY,1,-1
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) GO TO 290
            LL = (MINX(I) - 1) * 5 + 1
            WRITE(75,30) BB(1:LL),( POND(NODE), NODE = ISUM(I)+1, ISUM
     $      (I)+NPTS )
  290    CONTINUE
      ENDIF
c
      IF (initialize_annually_opt .EQ. 1) THEN
         DO I = MAXY,1,-1
            DO NODE = ISUM(I)+1,ISUM(I+1)
               HINIT(NODE) = H(NODE)
               POND0(NODE) = POND(NODE)
            END DO
         END DO
      ENDif
C-----------------
C  READ AQUIFER DEPTH VALUES   (FT FROM AQUIFER BOTTOM TO NGVD)
C-----------------
      IF (LP(5)) WRITE (75,300)
C
      DO 330 I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         NREC = (NDPTS-1)/NDATA + 1
         DO 310 J = 1,NREC
            READ(60,710) JY,IX1,IX2,( AQDEP(NODE), NODE = ISUM(JY)+IX1
```

```
     $      -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

  310    CONTINUE
         IF (.NOT.LP(5)) GO TO 330
         LL = (MINX(I) + NPTS - 15) * 5 + 1
         WRITE(75,320) B(1:LL),( AQDEP(NODE), NODE = ISUM(I)+1+NPTS,
     $   ISUM(I+1) )
  330 CONTINUE
C
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 340 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 340
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,320) BB(1:LL),( AQDEP(NODE), NODE = ISUM(I)+1, ISUM
     $      (I)+NPTS )
  340    CONTINUE
      ENDIF
C-----------------
      NDATA = 10
C
C  READ AQUIFER PERMEABILITY DATA   (FT/DAY * 10**-4)
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (LP(5)) WRITE (75,350)
C
      DO 390 I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         NREC = (NDPTS-1)/NDATA + 1
         DO 360 J = 1,NREC
            READ (60,730) JY,IX1,IX2,( TKX(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

  360    CONTINUE
C CONVERT TO ACTUAL UNITS: FT/DAY
         DO 370 JJ = ISUM(I)+1,ISUM(I+1)
            TKX(JJ) = TKX(JJ) * 10000.
  370    CONTINUE
         LL = (MINX(I) + NPTS - 15) * 5 + 1
         IF (LP(5)) WRITE (75,380) B(1:LL),( TKX(NODE)/10000., NODE =
     $   ISUM(I)+1+NPTS, ISUM(I+1) )
  390 CONTINUE
C
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 400 I = MAXY,1,-1
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) GO TO 400
            LL = (MINX(I) - 1) * 5 + 1
            WRITE(75,380) BB(1:LL),( TKX(NODE)/10000., NODE = ISUM(I)+1,
     $      ISUM(I)+NPTS )
  400    CONTINUE
      ENDIF
C-----------------
```

```
C  AQUIFER TRANSMISSIVITY ARRAY   (DEPTH * PERMEABILITY)
C-----------------
      IF (LP(5)) WRITE (75,410)
C
      IF (LP(5)) THEN
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
c
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c reduced precision in WRITE statement
            IF (LP(5)) WRITE (75,30) B(1:LL), ( (TKX(NODE)*AQDEP(NODE)
     $      /1000000.), NODE = ISUM(I)+1+NPTS, ISUM(I+1) )

         END DO
C
         WRITE (75,*) '  '
         DO 420 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 420
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
c reduced precision in WRITE statement
            WRITE(75,30) BB(1:LL),( (TKX(NODE)*AQDEP(NODE)/1000000.),
     $      NODE = ISUM(I)+1, ISUM(I)+NPTS )
  420    CONTINUE
      ENDIF
c-------------------------------------------------------
c<< Kris: one whole READ block added and
c   created READ for the new ET basins.
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      NDATA = 27
C
C  READ ET BASIN IDENTIFIERS   (IETZON ARRAY)
C-----------------
      IF (LP(5)) WRITE (75,430)
C
      DO 450 I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF(NPTS .LT. 0) NPTS = 0
         NREC  = (NDPTS-1)/NDATA + 1
         DO 440 J = 1,NREC
            READ(60,720) JY,IX1,IX2,( IETZON(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

  440    CONTINUE
         IF (.NOT.LP(5)) GO TO 450
         LL = (MINX(I) + NPTS - 15) * 5 + 1
         WRITE(75,700) B(1:LL),( IETZON(NODE), NODE = ISUM(I)+1+NPTS,
     $   ISUM(I+1) )
  450 CONTINUE
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 460 I = MAXY,1,-1
```

```
            IF (MINX(I) .GE. 15) GO TO 460
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,700) BB(1:LL),( IETZON(NODE), NODE = ISUM(I)+1,
     $      ISUM(I)+NPTS )
  460    CONTINUE
      ENDIF
c-------------------------------------------------------
c     set ietzon for each node = node number itself (temp fix).
c     the potet (reference) is calculated for each node based on weights
c     exception is lec (east of levees where old et basins are still maintained)
c     READ the pet weights assigned to each node based on inverse distance
c     squarted from the node to the 10 pet stations.
c-------------------------------------------------------
      DO I = 1,MAXY
         DO NODE = ISUM(I)+1, ISUM(I+1)
            READ (12,'(6X,20(F5.3,1X))') (pet_weight(node,k),k=1,
     $      netzon-1)
            ietzon(node) = node
         END DO
      END DO
c
C-----------------
      NDATA = 11
C
C  READ INFILTRATION RATE   (FT/DAY)
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (LP(5)) WRITE (75,470)
C
      DO I = MAXY,1,-1
         NDPTS = ISUM(I+1) - ISUM(I)
         NPTS = 15 - MINX(I)
         IF (MINX(I) .GE. 15) NPTS = 0
         NREC  = (NDPTS-1)/NDATA + 1
         DO J = 1,NREC
            READ(60,710) JY,IX1,IX2,( SINF(NODE), NODE = ISUM(JY)+IX1
     $      -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

         END DO
         IF (LP(5)) THEN
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            WRITE (75,30) B(1:LL),( SINF(NODE), NODE = ISUM(I)+1+NPTS,
     $      ISUM(I+1) )
         ENDIF
      END DO
      IF (LP(5)) THEN
         WRITE (75,*) '  '
         DO 480 I = MAXY,1,-1
            IF (MINX(I) .GE. 15) GO TO 480
            LL = (MINX(I) - 1) * 5 + 1
            NPTS = 15 - MINX(I)
            WRITE(75,30) BB(1:LL),( SINF(NODE), NODE = ISUM(I)+1, ISUM
     $      (I)+NPTS )
  480    CONTINUE
      ENDIF
c WRMI, 7/29/93,
```

```
c Read irrigated area by use for LECDA IF we're using LEC ET routine
      IF (use_lec_et .GE. 1) THEN
C-----------------
C  READ IRRIGATED AREAS FOR EACH OF THE 6 USE TYPES
C-----------------
c 1) urban_landscape irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,490)
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,1), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,1), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 500 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 500
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,1), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  500       CONTINUE
         ENDIF
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c 2) nursery irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,510)
C
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,2), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,2), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 520 I = MAXY,1,-1
```

```
               IF (MINX(I) .GE. 15) GO TO 520
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,2), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  520       CONTINUE
         ENDIF
C-----------------
c 3) golf course irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,530)
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,3), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,3), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 540 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 540
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,3), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  540       CONTINUE
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
c 4) agr. low-volume irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,550)
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,4), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,4), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
```

```
            WRITE (75,*) '  '
            DO 560 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 560
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,4), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  560       CONTINUE
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
c 5) agr. overhead irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,570)
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,5), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,5), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 580 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 580
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,5), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  580       CONTINUE
         ENDIF
C-----------------
c 6) agr. other irrigated acreage
C-----------------
         IF (LP(5)) WRITE (75,590)
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( irrig_area(NODE,6), NODE = ISUM
     $         (JY)+IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
            IF (LP(5)) WRITE(75,320) B(1:LL),( irrig_area(NODE,6), NODE
     $      = ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
```

```
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 600 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 600
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
               WRITE(75,320) BB(1:LL),( irrig_area(NODE,6), NODE = ISUM
     $         (I)+1, ISUM(I)+NPTS )
  600       CONTINUE
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  CALCULATE FRACTION OF IRRIGATED AREA FOR EACH CELL
C-----------------
         DO ia = 1,ntotres+n_add_lev_sys
            IF (nnodes(ia) .GT. 0) THEN
               DO n = 1,nnodes(ia)
                  DO itype = 1,num_use_type
                     irrig_area(node_reserv(ia,n),itype) = 0.0
                  ENDDO
               ENDDO
            ENDIF
         ENDDO
         DO i = 1,isum(maxy+1)
            frac_irr_area_t(i) = 0.0
            DO itype = 1,num_use_type
               frac_irr_area(i,itype) = irrig_area(i,itype)/ (
     $         (DX/5280.)*(DY/5280.) * 640. )
               frac_irr_area_t(i) = frac_irr_area_t(i) + frac_irr_area
     $         (i,itype)
            END DO
         END DO
C-----------------
C  READ EFFECTIVE ROOT ZONE DEPTH   (FT)
C-----------------
         IF (LP(5)) WRITE (75,610)
C
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( erzd(NODE), NODE = ISUM(JY)+IX1
     $         -MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
c convert values from inches to ft
c           DO jj = isum(i)+1,isum(i+1)
c              erzd(jj) = erzd(jj) / 12.
c           END do
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c different precision in WRITE statement
            IF (LP(5)) WRITE(75,620) B(1:LL),( erzd(NODE), NODE = ISUM
     $      (I)+1+NPTS, ISUM(I+1) )
         END DO
```

```
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 630 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 630
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
c different precision in WRITE statement
               WRITE(75,620) BB(1:LL),( erzd(NODE), NODE = ISUM(I)+1,
     $         ISUM(I)+NPTS )
  630       CONTINUE
         ENDIF
c
c   check for any grid cell that has irrigated area and root zone
c   depth is zero
c
         n_err_irrig = 0
         DO I = MAXY,1,-1
           DO KK = MINX(I),MAXX(I)
              NODECHK = KK - MINX(I) + 1 + ISUM(I)
              IF (erzd(NODECHK) .lt. 0.00001 .AND. frac_irr_area_t
     $        (NODECHK) .GT. 0.0001) THEN
                 n_err_irrig = n_err_irrig + 1
                 ixerr_irrig(n_err_irrig) = kk
                 iyerr_irrig(n_err_irrig) = i
              ENDIF
           ENDDO
         ENDDO
c
         IF (n_err_irrig .gt. 0) THEN
           DO irr = 1,n_err_irrig
             WRITE(*,1111) iyerr_irrig(irr),ixerr_irrig(irr)
 1111        FORMAT(/,'IRRIGATED AREA FOR ROW',2x,I3,2x,'COL',2x,
     $       I3,2x,'IS GREATER THAN ZERO BUT ROOT ZONE DEPTH IS ZERO. CH
     $ECK ERZD VALUES IN static_grid_values.dat FILE')
           ENDDO
           STOP
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  READ MAXIMUM SOIL MOISTURE HOLDING CAPACITY IN THE UNSATURATED ZONE   (AC-FT)
C----------------
         IF (LP(5)) WRITE (75,640)
C
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( smmax(NODE), NODE = ISUM(JY)
     $         +IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
c convert values from inches to ac-ft
c           DO jj = isum(i)+1,isum(i+1)
c              smmax(jj) = (smmax(jj) / 12.)*(4.*640.)
```

```
c           END do
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c different precision in WRITE statement
            IF (LP(5)) WRITE(75,620) B(1:LL),( smmax(NODE), NODE = ISUM
     $      (I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 650 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 650
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
c different precision in WRITE statement
               WRITE(75,620) BB(1:LL),( smmax(NODE), NODE = ISUM(I)+1,
     $         ISUM(I)+NPTS )
  650       CONTINUE
         ENDIF
C-----------------
C  READ FRACTION OF LANDSCAPE IRRIGATION RECEIVING WATER FROM PWS WELLS
C----------------
         IF (LP(5)) WRITE (75,660)
C
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( flirpws(NODE), NODE = ISUM(JY)
     $         +IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c
            IF (LP(5)) WRITE(75,380) B(1:LL),( flirpws(NODE), NODE =
     $      ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 670 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 670
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
c
               WRITE(75,380) BB(1:LL),( flirpws(NODE), NODE = ISUM(I)+1,
     $         ISUM(I)+NPTS )
  670       CONTINUE
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  READ FRACTION OF GOLF COURSE IRRIGATION RECEIVING WATER FROM TREATED
WASTEWATER
C----------------
         IF (LP(5)) WRITE (75,680)
C
```

```
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( fgirtww(NODE), NODE = ISUM(JY)
     $         +IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c
            IF (LP(5)) WRITE(75,380) B(1:LL),( fgirtww(NODE), NODE =
     $      ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 690 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 690
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
c
               WRITE(75,380) BB(1:LL),( fgirtww(NODE), NODE = ISUM(I)+1,
     $         ISUM(I)+NPTS )
  690       CONTINUE
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  READ FRACTION OF LANDSCAPE IRRIGATION RECEIVING WATER FROM TREATED WASTEWATER
C----------------
         IF (LP(5)) WRITE (75,695)
C
         DO I = MAXY,1,-1
            NDPTS = ISUM(I+1) - ISUM(I)
            NPTS = 15 - MINX(I)
            IF (MINX(I) .GE. 15) NPTS = 0
            NREC = (NDPTS-1)/NDATA + 1
            DO J = 1,NREC
               READ(60,710) JY,IX1,IX2,( flirtww(NODE), NODE = ISUM(JY)
     $         +IX1-MINX(JY)+1, ISUM(JY)+IX2-MINX(JY)+ 1)

            END DO
            LL = (MINX(I) + NPTS - 15) * 5 + 1
c
            IF (LP(5)) WRITE(75,380) B(1:LL),( flirtww(NODE), NODE =
     $      ISUM(I)+1+NPTS, ISUM(I+1) )
         END DO
C
         IF (LP(5)) THEN
            WRITE (75,*) '  '
            DO 697 I = MAXY,1,-1
               IF (MINX(I) .GE. 15) GO TO 697
               LL = (MINX(I) - 1) * 5 + 1
               NPTS = 15 - MINX(I)
c
               WRITE(75,380) BB(1:LL),( flirtww(NODE), NODE = ISUM(I)+1,
```

```
     $         ISUM(I)+NPTS )
  697       CONTINUE
         ENDIF
      ENDif
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
      CLOSE(60)
c
c initialize root zone depths
      DO i=1,isum(maxy+1)
c
         rzd(i) = amin1(erzd(i),ells(i)-h(i))
         IF (rzd(i).LE.0.)rzd(i) = 0.0
         IF (IBSN(i) .EQ. 43) THEN
            IF (frac_irr_area_t(i) .gt. 0.000001) THEN
               no_nonscane_irr_cells_l8 = no_nonscane_irr_cells_l8 + 1
            ENDIF
         ENDIF
      END DO
c
c initialize moisture content in the unsaturated zone
      DO 740 I = 1,max_ncells
         IF ((runmode .EQ. 'SIMUL' .AND. IBSN(I) .GE. 7 .AND.
     $   IBSN(I) .LE. 10 .AND. land_use_type(LUTYP(I)) .NE. 'WETLAND')
     $   .OR. IBSN(I) .EQ. 40 .OR. IBSN(I) .EQ. 10 .OR. IBSN(I)
     $   .EQ. 42) THEN
            SOLMX(I) = SOLINIT
         ELSE
            SOLMX(I) = 0.0
         ENDIF
         IF (initialize_annually_opt .EQ. 1) THEN
            SOLMX0(I) = SOLMX(I)
         ENDif
         sumrain(i) = 0.0
         sumet(i)   = 0.0
         ICURTAIN(I) = 0
  740 CONTINUE
c
      DO NODE = 1,max_ncells
         IF (LUTYP(NODE) .NE. 0) THEN
           OFML2(NODE) = OFML(LUTYP(NODE),1)
           rmin_ofml2(NODE) = rmin_ofml(LUTYP(NODE))
           OWPOND2(NODE) = OWPOND(LUTYP(NODE))
         ENDIF
      ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      RETURN
   10 FORMAT (1H1,T40,'LAND SURFACE ELEVATIONS   (FT NVGD)',/)
   30 FORMAT (A,27F5.2)
   60 FORMAT (1H1,T40,'SOIL STORAGE COEFFICIENT   (DIMENSIONLESS)',/)
   80 FORMAT (1H1,T40,'LAND USE INDICATORS',/)
  100 FORMAT (A,27(3X,I2))
  140 FORMAT (1H1,T40,'BASIN INDICATOR VALUES',/)
  190 FORMAT (1H1,T40,'INITIAL GROUNDWATER CONDITIONS:',3X,
     $'STAGE   (FT NGVD)',/)
  280 FORMAT(A,42F5.2)
  300 FORMAT (1H1,T40,'AQUIFER THICKNESS:',3X,'BOTTOM TO NGVD   (FT)',/)
```

```
320 FORMAT (A,27F5.0)
350 FORMAT (1H1,T40,'AQUIFER PERMEABILITY * 10**-4   (FT/DAY)',/)
380 FORMAT (A,27F5.3)
410 FORMAT (1H1,T40,'AQUIFER TRANSMISSIVITY (DEPTH * PERM',
   $'EABILITY * 10**-6)   (FT^2 / DAY)',/)
430 FORMAT (1H1,T40,'ET BASIN INDICATOR VALUES',/)
470 FORMAT (1H1,T40,'INFILTRATION RATE   (FT/DAY)',/)
490 FORMAT (1H1,T40,'LEC LANDSCAPE IRRIGATION ACREAGE        ',/)
510 FORMAT (1H1,T40,'LEC NURSERY IRRIGATED ACREAGE           ',/)
530 FORMAT (1H1,T40,'LEC GOLF COURSE IRRIGATED ACREAGE       ',/)
550 FORMAT (1H1,T40,'LEC AGR. LOW VOLUME IRRIGATED ACREAGE   ',/)
570 FORMAT (1H1,T40,'LEC AGR. OVERHEAD IRRIGATED ACREAGE     ',/)
590 FORMAT (1H1,T40,'LEC AGR. OTHER IRRIGATED ACREAGE        ',/)
610 FORMAT (1H1,T40,'EFFECTIVE ROOT ZONE DEPTH   (FT)',/)
620 FORMAT (A,27F5.1)
640 FORMAT (1H1,T40,
   $'UNSAT. ZONE MAX. SOIL MOISTURE HOLDING CAPACITY   (AC-FT)',/)
660 FORMAT (1H1,T40,
   $'FRACTION OF LANDSCAPE IRRIG RECEIVING WATER FROM PWS WELLS',/)
680 FORMAT (1H1,T40,
   $'FRACTION OF GOLF COURSE IRRIGATION FROM TREATED WASTEWATER',/)
695 FORMAT (1H1,T40,
   $'FRACTION OF LANDSCAPE IRRIGATION FROM TREATED WASTEWATER',/)
700 FORMAT (A,27(3X,I2))
710 FORMAT(3X,3I3,11F6.2)
720 FORMAT(3X,3I3,3X,27I2)
730 FORMAT(3X,3I3,10F6.3)
    END
```

gw_rchg_update.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: gw_rchg_update.F,v 1.11 2005/06/10 13:29:07 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/gw_rchg_update.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE UPDATES RECHARGE TERM (RCHG) OF GROUNDWATER
C   EQUATION DUE TO MUNICIPAL AND INDUSTRIAL WELLFIELD PUMPAGE
C   AND IRRIGATION FOR APPROPRIATE GRID CELLS
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE gw_rchg_update (wfld, ilkmg, imcnl, day_et_unsat,
     & day_et_poten, day_net_irr, day_et_unsat_irr, day_et_unsat_nonirr,
     & add_dmndagl8,struc_name_meas,il8)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'agdata.inc'

      DIMENSION day_et_poten(max_ncells,31),
     & day_et_unsat(max_ncells,31),
     & day_et_unsat_irr(max_ncells,31),
     & day_et_unsat_nonirr(max_ncells,31),
     & day_net_irr(max_ncells,max_n_type,31)

      CHARACTER wfld(max_n_wells)*9,struc_name_meas(max_n_struc_meas)*6

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C    INITIALIZE DAILY GW RECHARGE WITH AVERAGE  DAILY HISTORICAL PUMPAGE
C    FOR EACH MONTH
```

```
C  update RCHG() array with public water supply well pumpages

      DO 10 I=1,NWELL
         IF (WFLD(I) .NE. '50-00615-') THEN
            RCHG(LOCW(I)) = RCHG(LOCW(I)) - PUMP(I,MONTH) * 1.0E6 /
     $      (7.48 * GDAR * (1.0 - carfact(LOCW(I))))
         ELSE
            QU(ILKMG) = QU(ILKMG) - PUMP(I,MONTH) * 1.0E6 /(7.48)
         ENDIF
   10 CONTINUE

c  update rchg() array with industrial and residential self supplied
c  withdrawals

      DO i = 1,nwell_ind_rss
         rchg(loc_ind_rss(i)) = rchg(loc_ind_rss(i)) - pump_ind_rss
     $   (i,month) * 1.0E6 / (7.48 * GDAR * (1.0 - carfact
     $   (loc_ind_rss(i))))
      END DO
c
c execute this section IF we're using LEC ET
      IF (use_lec_et .NE. 0) THEN
c
c compute irrigation supplies for all irrigation use types and update recharge
c
c first map todays et components and nirr demands back to the original arrays
c
         DO lec_node = 1, num_lec_nodes
            model_node = model_node_in_lec(lec_node)
            L = model_node
            et_poten(L) = day_et_poten(L,iday)
            et_unsat(L) = day_et_unsat(L,iday)
            et_unsat_irr(L) = day_et_unsat_irr(L,iday)
            et_unsat_nonirr(L) = day_et_unsat_nonirr(L,iday)
            DO itype = 1,num_use_type
               net_irr(L,itype) = day_net_irr(L,itype,iday)
            END DO
         END DO
c
c compute DAILY values of irrigation supplies, total irrigation demands
c supplies and shortages, and daily adjusted unsat zone ET for all LECDA cells.
c
         dmndagmc = 0.0
         add_dmndagl8 = 0.0
         DO lec_node = 1, num_lec_nodes
            model_node = model_node_in_lec(lec_node)
            IF ( ibsn(model_node) .NE. 40 .AND. frac_irr_area_t
     $      (model_node) .GT. 0.) THEN
               CALL lec_et_set_cutbacks(lec_node)
c
c update RCHG() array with irrigation supplies taking into consideration
c fraction of landscape irrigation (use_type = 1) that is served by
c public water supply
               rchg(model_node) = rchg(model_node) - supply_t
     $         (model_node) + flirpws(model_node)*supply(model_node,1) +
     $         fgirtww(model_node)*supply(model_node,3) +
```

```
     $         flirtww(model_node)*supply(model_node,1)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c
c determine the irrigation supply from M-canal for appropriate grid
c cells (basin 41)
c
               IF (ibsn(model_node) .EQ. 41) THEN
                  dmndagmc = dmndagmc + ( supply_t(model_node)*gdar -
     $            flirpws(model_node)*supply(model_node,1)*gdar -
     $            fgirtww(model_node)*supply(model_node,3)*gdar -
     $            flirtww(model_node)*supply(model_node,1)*gdar) /
     $            86400.
               ENDIF
c
c determine the irrigation supply from L8 canal for other than sugar cane
c for appropriate cells (basin 43)
c
               IF (ibsn(model_node) .EQ. 43) THEN
                  add_dmndagl8 =  ( supply_t(model_node)*gdar -
     $            flirpws(model_node)*supply(model_node,1)*gdar -
     $            fgirtww(model_node)*supply(model_node,3)*gdar -
     $            flirtww(model_node)*supply(model_node,1)*gdar) /
     $            43560. + add_dmndagl8
               ENDIF
            ENDIF
         END DO
         qu(imcnl) = qu(imcnl) - amin1(dmndagmc,rmcagrfcap) * 86400.
         call match ('L8CP  ',1,6,struc_name_meas,nflpts,6,imatch)
         if (imatch .gt. 0) THEN
            qu(il8) = qu(il8) - add_dmndagl8*43560.
         endif
         IF (dmndagmc .GT. 0.0) THEN
            ratiosp   = amin1(rmcagrfcap/dmndagmc,1.0)
         ELSE
            ratiosp   = 1.0
         ENDIF
         DO lec_node = 1, num_lec_nodes
            model_node = model_node_in_lec(lec_node)
            IF (ibsn(model_node) .EQ. 41) THEN
               rchg(model_node) = rchg(model_node) + ( supply_t
     $         (model_node) - flirpws(model_node)*supply(model_node,1) -
     $         fgirtww(model_node)*supply(model_node,3) -
     $         flirtww(model_node)*supply(model_node,1) ) *ratiosp
               perc(model_node)=perc(model_node)+( supply_t(model_node)
     $          - flirpws(model_node)*supply(model_node,1) -
     $          fgirtww(model_node)*supply(model_node,3) -
     $          flirtww(model_node)*supply(model_node,1) )*ratiosp
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            ENDIF
            IF (ibsn(model_node) .EQ. 43) THEN
               rchg(model_node) = rchg(model_node) +  supply_t
     $         (model_node) - flirpws(model_node)*supply(model_node,1) -
     $         fgirtww(model_node)*supply(model_node,3) -
     $         flirtww(model_node)*supply(model_node,1)
               perc(model_node)=perc(model_node)+ supply_t(model_node)
     $         - flirpws(model_node)*supply(model_node,1) -
```

```
$         fgirtww(model_node)*supply(model_node,3) -
$         flirtww(model_node)*supply(model_node,1)
       ENDIF
    END DO
 ENDIF
 RETURN
 END
```

gwf.F

```
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: gwf.F,v 1.17 2005/06/10 13:29:07 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/gwf.F,v $
C   $Name: rel-5-5 $
C
C
C   PURPOSE OF SUBROUTINE IS TO SOLVE THE
C   TWO-DIMENSIONAL GROUNDWATER EQUATION USING
C   FINITE DIFFERENCE APPROACH FOR HEAD AT EACH
C   NODE. THE
C
C             VARIABLE DEFINITIONS   GROUNDWATER SUBROUTINE
C    AQDEP     THICKNESS OF AQUIFER BELOW MSL (FT)
C    A,B,C     SUB TERMS IN NUMERICAL EQUATION FOR H
C    D,E        "    "         "          "      "
C    H         GROUNDWATER STAGE AT EACH NODE
C    IXM1      SUBSCRIPT OF NODE IN X-1 DIRECTION
C    IXP1          "     "    "     X+1     "
C    IYM1          "     "    "     Y-1     "
C    IYP1          "     "    "     Y+1     "
C    MAXXT     MAXIMUM NUMBER OF NODES IN ANY ROW
C    NCA       (1,2,3,4) INDICATOR TO CONTROL ALT. DIR. SOLUTION
C    TKX       PERMEABILITY OF AQUIFER
C    TX ,TY    HARMONIC MEANS OF TRANSMISSIVITY
C
C-----------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE gwf (solmdph, jday, rchg_gw)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'c1.inc'
```

```
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'smallres.inc'

      DIMENSION gwhead(max_ncells), prevh(max_ncells),
     & rchg_gw(max_ncells), solmdph(max_ncells), temph(max_ncells),
     & tx(2), ty(2)

      INTEGER  x,y

      REAL elrz(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  ADD SURFACE PONDING DEPTH TO GW HEAD IF GW LEVEL IS EQUAL TO LAND
C  ELEVATION

      CRP = -2.0
      DO 10 I = MAXY,1,-1
         DO 10 J = ISUM(I) + 1,ISUM(I+1)
            IF (((IBSN(J) .LT. 7 .OR. IBSN(J) .GT. 10 .OR. land_use_type
     $      (LUTYP(J)) .EQ. 'WETLAND').AND.IBSN(J) .NE. 40 .AND.IBSN(J)
     $      .NE. 42) .OR. (IBSN(J) .NE. 10 .AND. IBSN(J) .NE. 40 .AND.
     $      runmode .EQ. 'CALIB')) THEN
C  NON-EAA CELL
               IF (use_lec_et * lec_node_in_model(j) .EQ. 0 .OR. ibsn(j)
     $         .EQ. 40 .OR. frac_irr_area_t(j) .EQ. 0.) THEN
C  original algorithm for non-eaa cell
                  IF (H(J) .GE. ELLS(J)) THEN
                     H(J) = H(J) + POND(J)
                     perc(j) = perc(j) + POND(J) * S(J)
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     infilt(j) = infilt(j) + POND(J) * S(J)
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     POND(J) = (1 - S(J)) * POND(J)
                  ENDIF
                  GWHEAD(J) = H(J)
clmb>>>
               ELSE
c*** revised algorithm for non-eaa cell
                  IF (erzd(j) .EQ. 0.) THEN
                     whc = 0.0
                  ELSE
                     whc = smmax(j) / ( (GDAR*(1.0-total_small_afact(j)
     $                -carfact(j))/43560.)*erzd(j) )
                  ENDIF
                  IF (H(J) .GE. ELLS(J)) THEN
c initial head for solution of gw equation includes additional head
c due to ponding
                     H(J) = H(J) + POND(J)
```

```
                     perc(j) = perc(j) + POND(J) * S(J)
     $               * (1.0 - total_small_afact(j) - carfact(j))
                     infilt(j) = infilt(j) + POND(J) * S(J)
     $               * (1.0 - total_small_afact(j) - carfact(j))
c pond() in the next line can be referred to as residual ponding
c it will be considered under the RESET CORRECT PONDING section
                     POND(J) = (1 - S(J)) * POND(J)
                     swscap(j) = 0.0
                  ELSE
                     uzdepth = amin1(erzd(j),ells(j)-h(j))
                     swscap(j) = whc*uzdepth
                  ENDIF
                  GWHEAD(J) = H(J)
               ENDIF
clmb<<<
            ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c******* IRRIGATED EAA CELL
               IF (runmode .EQ. 'SIMUL' .OR. IBSN(J) .EQ. 10 .OR.
     $         IBSN(J) .EQ.40) THEN
                  IF (IBSN(J) .GE. 7 .AND. IBSN(J) .LE. 10 .OR. IBSN(J)
     $            .EQ. 40 .OR. IBSN(J) .EQ. 42) THEN
                     IF (H(J) .GE. ELLS(J)-SOLMDPH(J)/S(J)) THEN
                        IF (H(J) + SOLMX(J)/S(J) .GE. ELLS(J)) THEN
                           H(J) = H(J) + SOLMX(J)/S(J) + POND(J)
                           POND(J) = (1 - S(J)) * POND(J)
                        ENDIF
                     ENDIF
                  ENDIF
                  TEMPH(J) = H(J)
                  GWHEAD(J) = H(J)
               ENDIF
            ENDIF
   10 CONTINUE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  BEGIN ALTERNATING DIRECTION EXPLICIT TECHNIQUE
C-----------------
      NCA = NCA + 1
      IF (NCA .GT. 4) NCA = 1
      GO TO(20, 20, 30, 30), NCA
   20 JJ = MAXY
      KK = MAXXT
      GO TO 40
   30 JJ = MAXXT
      KK = MAXY
C-----------------
   40 GGG = 0
      DO 210 JI = 1,JJ
         DO 210 KI = 1,KK
            GO TO (50,60,50,60),NCA
   50       J = JI
            K = KI
            GO TO 70
   60       J = JJ - JI + 1
            K = KK - KI + 1
C-----------------
```

```
   70       GO TO (80,80,90,90), NCA
   80       X = K
            Y = J
            GO TO 100
   90       X = J
            Y = K
  100       IF (X .LT. MINX(Y) .OR. X .GT. MAXX(Y)) GO TO 210
            NODE = X - MINX(Y) + 1 + ISUM(Y)
            DXR = DX
            DYR = DY
            IF (LUTYP(NODE) .EQ. 0) THEN
c-------------------------------------------------
c  subroutine get_tide( ) fetches the current cell's
c  tide for today.
c-------------------------------------------------
               H(NODE) = get_tide(y,jday,proj_sea_level_rise)
               GWHEAD(NODE) = 1.025* (H(NODE)-CRP) + CRP
               GO TO 210
            ENDIF
  110       IXM1 = NODE - 1
            IXP1 = NODE + 1
            IF (Y .GT. 1) IYM1 = X - MINX(Y-1) + 1 + ISUM(Y-1)
            IYP1 = X - MINX(Y+1) + 1 + ISUM(Y+1)
C-----------------
C  ESTABLISH BOUNDARY CONDITIONS FOR GROUND WATER CALCULATIONS.
C        -- CONSTANT HEAD AT COASTLINE
C        -- NO FLOW AT INTERIOR BOUNDARY NODES
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C   USE BLOCK IF CODING TO DETERMINE ACTIONS FOR BOUNDARY
C-----------------
  120       IF ( X .EQ. MINX(Y) ) THEN
               IXM1 = max_ncells-1
               TKX(IXM1) = TKX(NODE)
               AQDEP(IXM1) = AQDEP(NODE)
               GWHEAD(IXM1) = GWHEAD(NODE)
            ENDIF
C----------------
            JJYY = Y - 1
  130       IF (Y .EQ. 1) JJYY = 1
            IF (Y .EQ. 1 .OR. X .LT. MINX(JJYY)) THEN
               IYM1 = max_ncells
               TKX(IYM1) = TKX(NODE)
               AQDEP(IYM1) = AQDEP(NODE)
               GWHEAD (IYM1) = GWHEAD (NODE)
            ENDIF
            IF (X .GT. MAXX(JJYY)) THEN
               IYM1 = max_ncells
               TKX(IYM1) = TKX(NODE)
               AQDEP(IYM1) = AQDEP(NODE)
               H(IYM1) = get_tide(jjyy,jday,proj_sea_level_rise)
               GWHEAD(IYM1) = 1.025* (H(IYM1)-CRP) + CRP
            ENDIF
C-----------------
  140       IF (X .EQ. MAXX(Y)) THEN
               IXP1 = max_ncells - 1
               TKX(IXP1) = TKX(NODE)
```

```
               AQDEP(IXP1) = AQDEP(NODE)
               H(IXP1) = get_tide(y,jday,proj_sea_level_rise)
               GWHEAD(IXP1) = 1.025* (H(IXP1)-CRP) + CRP
            ENDIF
C-----------------
  150       IF (Y .EQ. MAXY .OR. X.LT.MINX(Y+1)) THEN
               IYP1 = max_ncells
               TKX (IYP1) = TKX(NODE)
               AQDEP(IYP1) = AQDEP(NODE)
               GWHEAD (IYP1) = GWHEAD(NODE)
            ENDIF
            IF (X .GT. MAXX(Y+1)) THEN
               IYP1 = max_ncells
               TKX(IYP1) = TKX(NODE)
               AQDEP(IYP1) = AQDEP(NODE)
               H(IYP1) = get_tide(y + 1,jday,proj_sea_level_rise)
               GWHEAD(IYP1) = 1.025* (H(IYP1)-CRP) + CRP
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
C  INCLUDE ANY PROPOSED (OR CURRENT) CURTAIN WALLS
C  CURTAIN WALLS SIMULATED AS NO FLOW CONDITION ACROSS
C  APPROPRIATE CELL INTERFACE(S)
C
            IF (ICURTAIN(NODE) .EQ. 1) THEN
               IF (BORIENT(NODE,1) .EQ. 'E'.AND. icurtw_opt
     $         (node,iseas,1) .EQ. 1) THEN
                  IXP1 = max_ncells - 3
                  GWHEAD(IXP1) = GWHEAD(NODE)
                  TKX(IXP1) = TKX(NODE)
                  AQDEP(IXP1) = AQDEP(NODE)
               ENDIF
               IF (BORIENT(NODE,2) .EQ. 'W'.AND. icurtw_opt
     $         (node,iseas,2) .EQ. 1) THEN
                  IXM1 = max_ncells - 2
                  GWHEAD(IXM1) = GWHEAD(NODE)
                  TKX(IXM1) = TKX(NODE)
                  AQDEP(IXM1) = AQDEP(NODE)
               ENDIF
               IF (BORIENT(NODE,3) .EQ. 'N'.AND. icurtw_opt
     $         (node,iseas,3) .EQ. 1) THEN
                  IYP1 = max_ncells - 1
                  GWHEAD(IYP1) = GWHEAD(NODE)
                  TKX(IYP1) = TKX(NODE)
                  AQDEP(IYP1) = AQDEP(NODE)
               ENDIF
               IF (BORIENT(NODE,4) .EQ. 'S'.AND.icurtw_opt
     $         (node,iseas,4) .EQ. 1) THEN
                  IYM1 = max_ncells
                  GWHEAD(IYM1) = GWHEAD(NODE)
                  TKX(IYM1) = TKX(NODE)
                  AQDEP(IYM1) = AQDEP(NODE)
               ENDIF
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  FIRST FIND CURRENT TRANSMISSIVITY VALUES FOR THESE GRID POINTS
```

```
C-----------------
  160       TH = TKX(NODE) * (AQDEP(NODE) + GWHEAD(NODE))
            XM = TKX(IXM1) * (AQDEP(IXM1) + GWHEAD(IXM1))
            XP = TKX(IXP1) * (AQDEP(IXP1) + GWHEAD(IXP1))
            YM = TKX(IYM1) * (AQDEP(IYM1) + GWHEAD(IYM1))
            YP = TKX(IYP1) * (AQDEP(IYP1) + GWHEAD(IYP1))
C-----------------
C  NOW CALCULATE HARMONIC MEAN OF THE TRANSMISSIVITIES
C-----------------
            TX(1) = (XM + TH) / (DXR * 2.)
            TX(2) = (XP + TH) / (DXR * 2.)
            TY(1) = (YM + TH) / (DYR * 2.)
            TY(2) = (YP + TH) / (DYR * 2.)
C-----------------
  170       A = TX(1) / DXR + TY(1) / DYR
            E = TX(2) / DXR + TY(2) / DYR
            GO TO (180,190,180,190), NCA
  180       A = A + S(NODE) / DT
            BBB = E
            GO TO 200
  190       BBB = A
            A = E + S(NODE) / DT
  200       C = S(NODE) * GWHEAD(NODE) / DT + RCHG(NODE)
            D = (TX(1) * GWHEAD(IXM1) + TX(2) * GWHEAD(IXP1)) / DXR +
     $      (TY(1) * GWHEAD(IYM1) + TY(2) * GWHEAD(IYP1)) / DYR
            PREVH(NODE) = GWHEAD(NODE)
            GWHEAD(NODE) = (C + D - BBB * GWHEAD(NODE)) / A
c           if (ibsn(node) .eq. 40) write(777,*)
c    $       month,iday,s(node),node,GWHEAD(NODE),RCHG(NODE),
c    $       PREVH(NODE),x,y,c,d,bbb,a,'   HMACKEREL ',tkx(ixp1),
c    $       tkx(node),tkx(iym1),tkx(ixm1),tkx(iyp1)
C
C     ESTIMATE GW FLOW VECTORS
C
            HAVG = (GWHEAD(NODE) + PREVH(NODE)) / 2.0
            H(NODE) = GWHEAD(NODE)
            HDXP1 = HAVG - GWHEAD(IXP1)
            HDXM1 = HAVG - GWHEAD(IXM1)
            HDYP1 = HAVG - GWHEAD(IYP1)
            HDYM1 = HAVG - GWHEAD(IYM1)
            IF (ICURTAIN(NODE) .EQ.  1) THEN
                IF (BORIENT(NODE,1) .EQ. 'E'.AND. icurtw_opt(node,
     $          iseas,1) .EQ. 1) THEN
                   HDXP1 = PREVH(NODE) - GWHEAD(IXP1)
                ENDIF
                IF (BORIENT(NODE,2) .EQ. 'W'.AND. icurtw_opt(node,
     $          iseas,2) .EQ. 1) THEN
                   HDXM1 = PREVH(NODE) - GWHEAD(IXM1)
                ENDIF
                IF (BORIENT(NODE,3) .EQ. 'N'.AND. icurtw_opt(node,
     $          iseas,3) .EQ. 1) THEN
                   HDYP1 = PREVH(NODE) - GWHEAD(IYP1)
                ENDIF
                IF (BORIENT(NODE,4) .EQ. 'S'.AND.icurtw_opt(node,
     $          iseas,4) .EQ. 1) THEN
                   HDYM1 = PREVH(NODE) - GWHEAD(IYM1)
                ENDIF
```

```
            ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
              TXPAV = (TH + XP) / 2.0
              TXMAV = (TH + XM) / 2.0
              TYPAV = (TH + YP) / 2.0
              TYMAV = (TH + YM) / 2.0
              DO ir = 1,noresincell(node)
                 ia  = ires_index_in_cell(node,ir)
                 IF (sfactor(ia) .GE. sfactmin .OR. ires_small_sim
     $           (ia) .eq. 'NO ') THEN
                    head_cell = est_cell_stg_o_res(node)
                    head_cell_avg = head_cell
                 ELSE
                    head_cell_avg = (gwhead(node)+PREVH(NODE))/2.0
                    head_cell = gwhead(node)
                 ENDIF

                 THC1 = TKX(NODE) * (AQDEP(NODE) + head_cell)
                 HDXCP1 = head_cell_avg - GWHEAD(IXP1)
                 HDXCM1 = head_cell_avg - GWHEAD(IXM1)
                 HDYCP1 = head_cell_avg - GWHEAD(IYP1)
                 HDYCM1 = head_cell_avg - GWHEAD(IYM1)
                 IF (ICURTAIN(NODE) .EQ.  1) THEN
                    IF (BORIENT(NODE,1) .EQ. 'E'.AND. icurtw_opt(node,
     $              iseas,1) .EQ. 1) THEN
                       HDXCP1 = PREVH(NODE) - GWHEAD(IXP1)
                    ENDIF
                    IF (BORIENT(NODE,2) .EQ. 'W'.AND. icurtw_opt(node,
     $              iseas,2) .EQ. 1) THEN
                       HDXCM1 = PREVH(NODE) - GWHEAD(IXM1)
                    ENDIF
                    IF (BORIENT(NODE,3) .EQ. 'N'.AND. icurtw_opt(node,
     $              iseas,3) .EQ. 1) THEN
                       HDYCP1 = PREVH(NODE) - GWHEAD(IYP1)
                    ENDIF
                    IF (BORIENT(NODE,4) .EQ. 'S'.AND.icurtw_opt(node,
     $              iseas,4) .EQ. 1) THEN
                       HDYCM1 = PREVH(NODE) - GWHEAD(IYM1)
                    ENDIF
                 ENDIF

                 TXPAV1 = (THC1 + XP) / 2.0
                 TXMAV1 = (THC1 + XM) / 2.0
                 TYPAV1 = (THC1 + YP) / 2.0
                 TYMAV1 = (THC1 + YM) / 2.0
                 hmvol1 = hdxcp1 * txpav1 * DYR * DT / DXR
                 hmvol2 = hdxcm1 * txmav1 * DYR * DT / DXR
                 hmvol3 = hdycp1 * typav1 * DXR * DT / DYR
                 hmvol4 = hdycm1 * tymav1 * DXR * DT / DYR
                 daily_gw_loss_cell(node) = (hmvol1 + hmvol2 +
     $           hmvol3 + hmvol4)/GDAR
              END DO
              HMVOL(NODE,1) = HMVOL(NODE,1)+ HDXP1 * TXPAV * DYR * DT /
     $        DXR
              HMVOL_daily(NODE,1) = HDXP1 * TXPAV * DYR * DT / DXR
```

```
                HMVOL(NODE,2) = HMVOL(NODE,2)+ HDYM1 * TYMAV * DXR * DT /
     $          DYR
                HMVOL_daily(NODE,2) = HDYM1 * TYMAV * DXR * DT / DYR

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c             IF (ICURTAIN(NODE) .EQ. 1) THEN
c               print *,HMVOL(NODE,1)/43560.,HMVOL(NODE,2)/43560.
c    $               ,BORIENT(NODE,1),BORIENT(NODE,2),BORIENT(NODE,3)
c    $               ,BORIENT(NODE,4),X,Y,MONTH,IDAY,NCA
c             ENDIF
C
  210 CONTINUE
c
c  adjust water level in cells with small reservoirs for regional gw flow
c
c     nodebf = 23-minx(44)+1+isum(44)
      DO 310 I = MAXY,1,-1
         DO 310 J = MINX(I),MAXX(I)
            NODE = J - MINX(I) + 1 + ISUM(I)
            IF (total_small_afact(NODE) .gt. 0.000001 .or. carfact(node)
     $         .gt. 0.000001) THEN
                NXM1 = NODE - 1
                NYP1 = J - MINX(I+1) + 1 + ISUM(I+1)
                RNETGW = HMVOL_daily(NODE,1) + HMVOL_daily(NODE,2)
     $          - HMVOL_daily(NXM1,1) - HMVOL_daily(NYP1,2)
                H(NODE) = H(NODE) - ((RNETGW/GDAR)
     $          *((total_small_afact(node)+carfact(node))/(1.0-
     $          total_small_afact(node)-carfact(node))))/s(node)
            ENDIF
  310 CONTINUE

C-----------------
C  RESET CORRECT PONDING WHERE H IS ABOVE LAND SURFACE OR ROOT ZONE
C-----------------
      DO 230 I = MAXY,1,-1
         DO 230 J = ISUM(I) +1,ISUM(I+1)
            rchg_gw(J) = RCHG(J)
            RCHG(J) = 0.0
            IF(((IBSN(J) .LT. 7 .OR. IBSN(J) .GT. 10 .OR. land_use_type
     $      (LUTYP (J)) .EQ. 'WETLAND') .AND. IBSN(J) .NE. 40 .AND.
     $      IBSN(J) .NE. 42) .OR. (IBSN(J) .NE. 10 .AND. IBSN(J) .NE.
     $      40 .AND. runmode .EQ. 'CALIB')) THEN
c******* NON-EAA CELL
clmb>>>
                IF (use_lec_et * lec_node_in_model(j) .EQ. 0 .OR. ibsn(j)
     $          .EQ. 40 .OR. frac_irr_area_t(j) .EQ. 0.) THEN
c*** original algorithm for non-eaa cell
                   IF (H(J) .GE. ELLS(J)) THEN
c resulting stage above ground
                      IF (lutyp(j) .NE. 0) THEN
                         POND(J) = POND(J) + (H(J) - ELLS(J)) * S(J)
                         perc(j) = perc(j) - (H(J) - ELLS(J)) * S(J)
     $                   * (1.0 - total_small_afact(j)-carfact(j))
                         infilt(j) = infilt(j) - (H(J) - ELLS(J)) * S(J)
     $                   * (1.0 - total_small_afact(j) - carfact(j))
                         H(J) = ELLS(J)
                      ELSE
```

```
                        pond(j) = h(j) - ells(j)
                        h(j) = ells(j)
                     ENDIF
                  ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c resulting stage below ground
c FILIM = available void in soil
                     FILIM = (ELLS(J) - H(J)) * S(J)
c FLMAX = max infilt rate * 1 day
                     FLMAX = SINF(J)
c AVINF = available space for "residual" ponding to infiltrate
                     AVINF = AMIN1(FLMAX,FILIM)
                     IF (POND(J) .GE. AVINF)  THEN
c residual ponding is enough to saturate the whole soil column
clmb                      H(J) = ELLS(J)
                        H(J) = H(J) + AVINF/S(J)
                        perc(j) = perc(j) + avinf*(1.0-total_small_afact
     $                  (j)-carfact(j))
                        infilt(j) = infilt(j) + avinf*(1.0 -
     $                  total_small_afact(j)-carfact(j))
                        POND(J) = POND(J) - AVINF
                     ELSE
c residual ponding is not enough to saturate the whole soil column
                        H(J) = H(J) + POND(J) / S(J)
                        perc(j) = perc(j) + pond(j)*(1.0-
     $                  total_small_afact(j)-carfact(j))
                        infilt(j) = infilt(j) + pond(j)*(1.0 -
     $                  total_small_afact(j)-carfact(j))
                        POND(J) = 0.0
                     ENDIF
                  ENDIF
               ELSE
c*** revised algorithm for non-eaa cell
c elrz(j) = elevation of bottom of full unsaturated zone (= eff. root zone depth
                  elrz(j) = ells(j) - erzd(j)
                  IF (erzd(j) .EQ. 0.) THEN
                     whc = 0.0
                  ELSE
                     whc = smmax(j) / ( (GDAR*(1.0-total_small_afact(j)
     $                -carfact(j))/43560.)*erzd(j) )
                  ENDIF
                  IF (h(j) .GE. ells(j))THEN
c resulting stage above ground
                     IF (lutyp(j) .NE. 0) THEN
                        POND(J) = POND(J) + (H(J) - ELLS(J)) * S(J) +
     $                  SOLMX(J)
                        perc(j) = perc(j) - (H(J)-ELLS(J))*S(J)
     $                  *(1.0-total_small_afact(j)-carfact(j))
                        infilt(j) = infilt(j) - ( (H(J)-ELLS(J))*S(J) +
     $                  solmx(j) )*(1.0-total_small_afact(j)-carfact(j))
                        H(J) = ELLS(J)
                     ELSE
                        pond(j) = h(j) - ells(j)
                        h(j) = ells(j)
                     ENDIF
                     solmx(j) = 0.
                     rzd(j) = 0.
```

```
                    swscap(j) = 0.
                 ELSE
c resulting stage below ground
                    FILIM = (ELLS(J) - H(J)) * S(J) - solmx(j)
                    IF (FILIM .GT. 0.) THEN
clmb case:  FILIM > 0.0; there is available void in soil
                       FLMAX = SINF(J)
                       AVINF = AMIN1(FLMAX,FILIM)
                       IF(POND(J) .GE. AVINF) THEN
c residual ponding is enough to saturate the entire soil column
c also, unsaturated zone thickness will be zero
                          IF (AVINF .EQ. FLMAX) THEN
                             H(J) = H(J) + (AVINF+solmx(j))/S(J)
                          ELSE
                             H(J) = ELLS(J)
                          ENDIF
                          POND(J) = POND(J) - AVINF
c note: solmx() and pond() may both be positive before the
c       gw flow equations are solved. Thus, updat infilt() and
c       perc() for ponding, and perc(), again, for solmx()
c
                          perc(j) = perc(j) + (avinf + solmx(j))
     $                     * (1.0-total_small_afact(j)-carfact(j))
                          infilt(j) = infilt(j) + avinf
     $                     * (1.0-total_small_afact(j)-carfact(j))
clmblmb                    SINFLTM(J) = SINFLTM(J) + AVINF*12
                          solmx(j) = 0.
                          rzd(j) = 0.
                          swscap(j) = 0.
                       ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c residual ponding is not enough to saturate the entire soil column,
c either because a) it is zero to start with (+ solmx(j) >= 0.) OR
c b) AVINF > POND(J)
c procedure: initially, assume all available space is available for
c full saturation THEN, IF it falls below the bottom of the full
c unsaturated zone (or resulting h() does not push solmx() to exceed water
c holding capacity of of resulting unsat. zone depth) ==> h() is set
c and solmx() is unchanged; ELSE ==> iterate for h()
                          H(J) = H(J) + POND(J)/S(J)
                          perc(j) = perc(j) + pond(j)
     $                     * (1.0-total_small_afact(j)-carfact(j))
                          infilt(j) = infilt(j) + pond(j)
     $                     * (1.0-total_small_afact(j)-carfact(j))
clmblmb                    SINFLTM(J) = SINFLTM(J) + POND(J)*12
                          POND(J) = 0.0
                          IF (h(j) .LE. elrz(j)) THEN
c solmx(j) is unchanged; all of residual ponding went to saturated zone
c current root zone depth is at maximum; water table rise due to residual
c ponding did not encroach upon effective root zone depth
                             rzd(j) = erzd(j)
                             swscap(j) = rzd(j)*whc
                          ELSE
c solmx(j) may change; water table encroached upon effective root zone
c depth; current root zone depth is not at its maximum
                             tempswscap = (ells(j) - h(j))*whc
                             IF (solmx(j) .LE. tempswscap) THEN
```

```
c solmx(j) is unchanged; moisture in the original unsat. zone does not
c exceed the water holding capacity of the resulting unsaturated zone
c i.e. water table rise is not enough to "compress" soil moisture to
c exceed whc
                                rzd(j) = ells(j) - h(j)
                                swscap(j) = tempswscap
                             ELSE
c solmx(j) will be changed, i.e., some of the moisture in the original
c unsaturated zone will drain to the saturated zone since the water
c holding capacity of the resulting unsaturated zone is less than
c the moisture content in the original unsaturated zone; the
c volume exchange is between the saturated and unsaturated zones
c only, therefore deal with percolation updates only
c
c DO iterative procedure to determine final water table location
c some of solmx() remains in the root zone and the remainder ENDs
c up as additional percolation
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
  220                           add_perc = solmx(j) - tempswscap
                                perc(j) = perc(j) + add_perc
     $                          * (1.0-total_small_afact(j)-carfact(j))
                                solmx(j) = tempswscap
                                h(j) = h(j) + add_perc/s(j)
                                rzd_new = ells(j) - h(j)
                                tempswscap = rzd_new*whc
                                IF (solmx(j) .LT. tempswscap-0.00001)
     $                          THEN
                                   WRITE(*,*) 'NODE:  ',j
                                   WRITE(*,*) 'YEAR: ',IYEAR+IFYR-1
                                   WRITE(*,*) 'MONTH: ',MONTH
                                   WRITE(*,*) 'IDAY:  ',IDAY
                                   CONTINUE
                                   WRITE(*,*)
     $                             'Something is wrong here!!!'
                                   WRITE(*,*) h(j),rzd_new,add_perc,
     $                             solmx(j) ,tempswscap

                                   stop
                                ELSE
                                   IF ((solmx(j)-tempswscap) .GT.
     $                             0.0001) GO TO 220
                                ENDIF
                                rzd(j) = rzd_new
                                swscap(j) = tempswscap
                                solmx(j) = swscap(j)
                             ENDIF
                          ENDIF
                       ENDIF
                    ELSE
clmb case:  FILIM <= 0.0; there is no available void in the soil;
c soil moisture in the previous unsaturated zone is enough to saturate
c the entire soil column and possibly increase ponding
                       POND(J) = POND(J) + abs(FILIM)
                       perc(j) = perc(j) + (ELLS(J)-H(J))*S(J)
     $                 *(1.0-total_small_afact(j)-carfact(j))
                       infilt(j) = infilt(j) - ( solmx(j) - (ELLS(J)-H
     $                 (J))*S(J))*(1.0-total_small_afact(j)-carfact(j))
```

```
                      H(J) = ELLS(J)
                      solmx(j) = 0.0
                      rzd(j) = 0.
                      swscap(j) = 0.
                   ENDIF
                ENDIF
             ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
          ELSE
c******* EAA CELL
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             IF (H(J) .GE. ELLS(J)) THEN
                SOLMX(J) = SOLMDPH(J)
                POND(J)  = POND(J) + (H(J) - ELLS(J)) * S(J)
                H(J) = ELLS(J) - SOLMDPH(J)/S(J)
             ELSE
                IF (TEMPH(J) .GE. ELLS(J)) THEN
                   FILIM = (ELLS(J) - H(J)) * S(J)
                   FLMAX = SINF(J)
                   AVINF = AMIN1(FLMAX,FILIM)
                   IF (POND(J) .GE. AVINF) THEN
                      H(J) = ELLS(J)-SOLMDPH(J)/S(J)
                      POND(J) = POND(J) - AVINF
                      SOLMX(J) = SOLMDPH(J) - (FILIM-AVINF)
                   ELSE
                      H(J) = H(J) + POND(J)/S(J)
                      SOLMX(J) = (H(J)-ELLS(J)+SOLMDPH(J)/S(J))
     $                * S(J)
                      POND(J) = 0.0
                      H(J) = ELLS(J) - SOLMDPH(J)/S(J)
                   ENDIF
                ELSEIF (TEMPH(J) .GE. ELLS(J)-SOLMDPH(J)/S(J)) THEN
                   ADDPTH = (ELLS(J)-SOLMDPH(J)/S(J)-H(J))*S(J)
                   ADDPTH = AMAX1(ADDPTH,0.0)
                   ADJDPTH = AMIN1(POND(J),ADDPTH)
                   POND(J) = POND(J) - ADJDPTH
                   DPHDFF = ADDPTH - ADJDPTH
                   ADJMX = AMIN1(SOLMX(J),DPHDFF)
                   ADJMX = AMAX1(ADJMX,0.0)
                   SOLMX(J) = SOLMX(J) - ADJMX
                   H(J) = H(J) + (ADJMX + ADJDPTH)/S(J)
                   IF (H(J) .GE. ELLS(J)-SOLMDPH(J)/S(J)) THEN
                      SOLMX(J) = SOLMX(J) + (H(J)-ELLS(J)+SOLMDPH(J) /
     $                S(J)) * S(J)
                      POND(J) = AMAX1(SOLMX(J)-SOLMDPH(J),0.0)
                      SOLMX(J) = SOLMX(J) - POND(J)
                      H(J) = ELLS(J) - SOLMDPH(J)/S(J)
                   ENDIF
                ELSEIF (H(J) .GE. ELLS(J)-SOLMDPH(J)/S(J)) THEN
                   SOLMX(J) =SOLMX(J) + (H(J)-ELLS(J)+SOLMDPH(J)/S(J))
     $             * S(J)
                   POND(J) = AMAX1(SOLMX(J)-SOLMDPH(J),0.0)
                   SOLMX(J) = SOLMX(J) - POND(J)
                   H(J) = ELLS(J) - SOLMDPH(J)/S(J)
                ELSE
                   ADDPTH = (ELLS(J)-SOLMDPH(J)/S(J)-H(J))*S(J)
```

```
                    ADJDPTH = AMIN1(POND(J),ADDPTH)
                    POND(J) = POND(J) - ADJDPTH
                    DPHDFF  = ADDPTH - ADJDPTH
                    ADJMX   = AMIN1(SOLMX(J)-SOLCRNF(J,JDAY),DPHDFF)
                    ADJMX   = AMAX1(ADJMX,0.0)
                    SOLMX(J) = SOLMX(J) - ADJMX
                    H(J)  = H(J) + (ADJMX + ADJDPTH)/S(J)
                 ENDIF
              ENDIF
           ENDIF
  230 CONTINUE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      RETURN
      END
```

hist_flow_adjust.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: hist_flow_adjust.F,v 1.13 2005/06/10 13:29:08 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/hist_flow_adjust.F,v $
C   $Name: rel-5-5 $
C
C
C SUBROUTINE ADJUSTS FLOWS PROVIDED BY DATA BASE WHEN NECESSARY. ONLY DONE
DURING
C  CALIBRATION/VERIFICATION
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE hist_flow_adjust (struc_name_meas, agrunoff_to_l8,
     & c304_out_to_marsh, prevqu)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'simflows.inc'

      DIMENSION prevqu(max_ncnls)

      CHARACTER struc_name_meas(max_n_struc_meas)*6

      LOGICAL drwdown

      DOUBLE PRECISION l38stg_min, dnstg, upstg, dble14
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      DO i=1,nflpts

        CALL match (struc_name_meas(i), 1, 6, flnm, 300, 6, imatch_s)
        IF (flnm(imatch_s).eq.'S11   ') THEN

          s11_flow=kflo(imatch_s)
          drwdown=ifyr+iyear-1.gt.1979.and.ifyr+iyear-1.le.1987
          l38stg_min=10.75
          IF (drwdown) THEN
            IF (s11_flow.gt.2000.) THEN
              s11_flow=2000.+(s11_flow-2000.)*0.75
            END IF
            l38stg_min=9.5
          ELSE
            IF (s11_flow.gt.2000..and.ifyr+iyear-1.gt.1991.and.ifyr+
     &       iyear-1.le.1993) THEN
              s11_flow=2000.+(s11_flow-2000.)*2.00
            END IF
          END IF
C
          avvol=(chdep(il38)-l38stg_min)*cl(il38)*width(il38)+
     &     cgseep(il38,l38stg_min,0,0.)+tovflc(il38,l38stg_min,0,0.,0,1)
     &     -(kflo(ifls38)+kflo(ifls143)+kflo(ifls144)+kflo(ifls145)+
     &     kflo(ifls146))*86400.
          avvol=amax1(avvol,0.0)/86400.
          tot_hist_reg=s11_flow
          IF (tot_hist_reg.gt.0.0) THEN
            ratio=avvol/tot_hist_reg
            ratio=amin1(ratio,1.0)
          ELSE
            ratio=1.0
          END IF
C
          s11_flow=ratio*s11_flow
C
          kflo(imatch_s)=s11_flow
          jstr=jstr+1
          strflw(jstr,idss)=kflo(imatch_s)
        ELSE IF (flnm(imatch_s).eq.'S337  ') THEN
          CALL match ('C304 ', 1, 5, cnm, nch, 5, ic304)
          dnstg=max(chdep(l30)+hdc(l30,iseas),3.5)
          avvol=(chdep(ic304)-dnstg)*cl(ic304)*width(ic304)+
     &     cgseep(ic304,dnstg,0,0.)+tovflc(ic304,dnstg,0,0.,0,1)+
     &     (kflo(ifls151)-kflo(ifls31)-c304_out_to_marsh)*86400.
          avvol=amax1(avvol,0.0)/86400.
          s337_flow=amin1(kflo(imatch_s),avvol)
          kflo(imatch_s)=s337_flow
          jstr=jstr+1
          strflw(jstr,idss)=kflo(imatch_s)
        ELSE IF (flnm(imatch_s).eq.'S339 ') THEN
          CALL match ('L23E ', 1, 5, cnm, nch, 5, il23e)
          CALL match ('C123 ', 1, 5, cnm, nch, 5, ic123)
          dnstg=chdep(ic123)+hdc(ic123,iseas) + 0.05
c
c    10.6 - mean lsel along L23E
```

```
c
c          avail_cstor = (10.6-chdep(il23e))
c    &     * cl(il23e)*width(il23e) - max(tovflc(il23e
c    &     ,10.6,0,0.,0,1),0.0) - cgseep(il23e,10.6,0,0.)
c          avail_cstor = amax1(avail_cstor,0.0)/86400.
c          if (avail_cstor .lt. 0.00001) then
c            if (chdep(il23e) .gt. 11.10) then
c               up_flow_avail = kflo(ifls8)
c            else
c               up_flow_avail = 0.0
c            endif
c          else
c            up_flow_avail = kflo(ifls8)
c          endif
           avvol=(chdep(il23e)-dnstg)*cl(il23e)*width(il23e)+
     &     cgseep(il23e,dnstg,0,0.)+tovflc(il23e,dnstg,0,0.,0,1)+
     &     kflo(ifls8)*86400.
           avvol=amax1(avvol,0.0)/86400.
           s339_flow=amin1(avvol,kflo(imatch_s))
           kflo(imatch_s)=s339_flow
           jstr=jstr+1
           strflw(jstr,idss)=kflo(imatch_s)
         ELSE IF (flnm(imatch_s).eq.'S340 ') THEN
           CALL match ('C123 ', 1, 5, cnm, nch, 5, ic123)
           dnstg=chdep(ica3)+0.7*hdc(ica3,iseas)
           avvol=(chdep(ic123)-dnstg-0.01)*cl(ic123)*width(ic123)+
     &     cgseep(ic123,dnstg,0,0.)+tovflc(ic123,dnstg,0,0.,0,1)+
     &     kflo(ifls339)*86400.
           avvol=amax1(avvol,0.0)/86400.
           s340_flow=amin1(avvol,kflo(imatch_s))
           kflo(imatch_s)=s340_flow
           jstr=jstr+1
           strflw(jstr,idss)=kflo(imatch_s)
         ELSE IF (flnm(imatch_s).eq.'S140  ') THEN
           CALL match ('L4   ', 1, 5, cnm, nch, 5, il4)
           upstg=8.5
           avvol=(chdep(il4)-upstg)*cl(il4)*width(il4)+cgseep(il4,upstg,
     &     0,0.)+tovflc(il4,upstg,0,0.,0,1)+kflo(iflg89)*86400.
           avvol=amax1(avvol,0.0)/86400.
           s140_flow=amin1(avvol,kflo(imatch_s))
           kflo(imatch_s)=s140_flow
           jstr=jstr+1
           strflw(jstr,idss)=kflo(imatch_s)
         ELSE IF (flnm(imatch_s).eq.'S175 ') THEN
           CALL match ('S175D ', 1, 5, cnm, nch, 5, is175d)
           dnstg=chdep(is175d)+hdc(is175d,iseas)
           avvol=(chdep(l31w)-dnstg-0.01)*cl(l31w)*width(l31w)+
     &     cgseep(l31w,dnstg,0,0.)+tovflc(l31w,dnstg,0,0.,0,1)+
     &     (kflo(ifls174)-kflo(ifls332))*86400.
           avvol=amax1(avvol,0.0)/86400.
           s175_flow=amin1(avvol,kflo(imatch_s))
           kflo(imatch_s)=s175_flow
           jstr=jstr+1
           strflw(jstr,idss)=kflo(imatch_s)
         ELSE IF (flnm(imatch_s).eq.'S197 ') THEN
           CALL match ('S197  ', 1, 5, cnm, nch, 5, is197u)
           dnstg=0.8
```

```
      avvol=(chdep(is197u)-dnstg)*cl(is197u)*width(is197u)+
&      cgseep(is197u,dnstg,0,0.)+tovflc(is197u,dnstg,0,0.,0,1)+
&      prevqu(is197u)
      avvol=amax1(avvol,0.0)/86400.
      s197_flow=amin1(avvol,kflo(imatch_s))
      kflo(imatch_s)=s197_flow
      jstr=jstr+1
      strflw(jstr,idss)=kflo(imatch_s)
    ELSE IF (flnm(imatch_s).eq.'S10  ') THEN
      CALL match ('S10E  ', 1, 6, flnm, 300, 6, ifls10e)
      dble14 = dble(14.0)
      avvol=(chdep(ica1)-dble14)*cl(ica1)*width(ica1)+cgseep(ica1,
&      dble14,0,0.)+tovflc(ica1,dble14,0,0.,0,1)+(kflo(ifls5a5)-
&      kflo(ifls39)-kflo(ifls10e))*86400.
      avvol=amax1(avvol,0.0)/86400.
      s10_flow=amin1(avvol,kflo(imatch_s))
      kflo(imatch_s)=s10_flow
      jstr=jstr+1
      strflw(jstr,idss)=kflo(imatch_s)
    ELSE IF (flnm(imatch_s).eq.'S332  ') THEN
      upstg=1.0
      avvol=(chdep(l31w)-upstg)*cl(l31w)*width(l31w)+cgseep(l31w,
&      upstg,0,0.)+tovflc(l31w,upstg,0,0.,0,1)+kflo(ifls174)*86400.
      avvol=amax1(avvol,0.0)/86400.
      s332_flow=amin1(avvol,kflo(imatch_s))
      kflo(imatch_s)=s332_flow
      jstr=jstr+1
      strflw(jstr,idss)=kflo(imatch_s)
    ELSE IF (flnm(imatch_s).eq.'S38  ') THEN
      drwdown=ifyr+iyear-1.gt.1979.and.ifyr+iyear-1.le.1987
      IF (drwdown) THEN
        upstg=8.0
      ELSE
        upstg=9.0
      END IF
      IF (nint(kflo(ifls11)).eq.0.) THEN
        avvol=(chdep(il38)-upstg)*cl(il38)*width(il38)+kflo(ifls7)*
&        86400.-(kflo(ifls143)+kflo(ifls144)+kflo(ifls145)+
&        kflo(ifls146))*86400.+tovflc(il38,upstg,0,0.,0,1)+
&        cgseep(il38,upstg,0,0.)
        avvol=amax1(avvol,0.0)/86400.
        kflo(imatch_s)=amin1(avvol,kflo(imatch_s))
      END IF
      jstr=jstr+1
      strflw(jstr,idss)=kflo(imatch_s)
    ELSE IF (flnm(imatch_s).eq.'S9   ') THEN
      CALL match ('C11W ', 1, 5, cnm, nch, 5, ic11w)
      upstg=0.0
      avvol=(chdep(ic11w)-upstg)*cl(ic11w)*width(ic11w)+
&      cgseep(ic11w,upstg,0,0.)+tovflc(ic11w,upstg,0,0.,0,1)+
&      qu(ic11w) + 200.*86400.
      avvol=amax1(avvol,0.0)/86400.
      s9_flow=amin1(avvol,kflo(imatch_s))
      kflo(imatch_s)=s9_flow
      jstr=jstr+1
      strflw(jstr,idss)=kflo(imatch_s)
    ELSE IF (flnm(imatch_s).eq.'S5A3  ') THEN
```

```
          upstg=13.0
C following lines commented out in v5.0 since this code is not used
C unless in calib mode and s316 is online
C
C             if (opt_l8_prop) then
C               il8c = il8s
C               agrunoff_to_l8c = agrunoff_to_l8s
C              upstrm_infl = outf_from_l8
C             else
          il8c=il8
          agrunoff_to_l8c=agrunoff_to_l8
          upstrm_infl=kflo(iflc10a)
C             endif
          IF (nint(kflo(imatch_s)).gt.0) THEN
            avvol=(chdep(il8c)-upstg)*cl(il8c)*width(il8c)+cgseep(il8c,
     &       upstg,0,0.)+tovflc(il8c,upstg,0,0.,0,1)+(upstrm_infl-
     &       kflo(ifls2mc)+agrunoff_to_l8c)*86400.
            avvol=amax1(avvol,0.0)/86400.
            rl8_flow=amin1(avvol,kflo(imatch_s))
            rl8_flow_orig=kflo(imatch_s)
            rlim_flow_adj=rl8_flow_orig-rl8_flow-amax1(kflo(iflbel5),
     &       0.0)
            rlim_flow_adj=amax1(rlim_flow_adj,0.0)
            kflo(imatch_s)=rl8_flow+amax1(rlim_flow_adj-
     &       amax1(kflo(ifls5a5),0.0),0.0)
            IF (nint(kflo(iflbel5)).gt.0) THEN
              kflo(iflbel5)=kflo(iflbel5)-amin1(rl8_flow_orig-
     &         kflo(imatch_s),kflo(iflbel5))
            END IF
            IF (nint(kflo(ifls5a5)).gt.0) THEN
              kflo(ifls5a5)=kflo(ifls5a5)-amin1(rlim_flow_adj,
     &         kflo(ifls5a5))
            END IF
          END IF
          jstr=jstr+1
          strflw(jstr,idss)=kflo(ifls5a5)
        ELSE IF (flnm(imatch_s).eq.'S5A4  ') THEN
          IF (kflo(imatch_s).lt.-0.00001) THEN
            upstg=10.0
            upstrm_infl=prevqu(ic51w)
            avvol=(chdep(ic51w)-upstg)*cl(ic51w)*width(ic51w)+
     &       cgseep(ic51w,upstg,0,0.)+tovflc(ic51w,upstg,0,0.,0,1)+
     &       upstrm_infl
            avvol=amax1(avvol,0.0)/86400.
            s5a4_flw=abs(kflo(imatch_s))
            IF (nint(kflo(ifls5a5)).gt.0.and.nint(kflo(imatch_s))
     &       .ne.nint(kflo(ifll8))) THEN
              rlim_flow_adj=amax1(s5a4_flw-avvol,0.0)
              rlim_flow_adj=amin1(rlim_flow_adj,amax1(kflo(ifls5a5),0.0)
     &         )
              kflo(imatch_s)=-s5a4_flw+rlim_flow_adj
              IF (nint(kflo(ifls5a5)).gt.0) THEN
                kflo(ifls5a5)=kflo(ifls5a5)-rlim_flow_adj
              END IF
            ELSE
              IF (kflo(ifll8).lt.-0.00001) THEN
                rlim_flow_adj=amax1(s5a4_flw-avvol,0.0)
```

```
          rlim_flow_adj=amin1(rlim_flow_adj,abs(kflo(ifll8)))
          kflo(imatch_s)=-s5a4_flw+rlim_flow_adj
          kflo(ifll8)=kflo(ifll8)+rlim_flow_adj
        END IF
      END IF
    END IF
    jstr=jstr+1
    strflw(jstr,idss)=kflo(imatch_s)
    jstr=jstr+1
    strflw(jstr,idss)=kflo(ifll8)
  END IF
END DO
RETURN
END
```

hist_lok_bndry_input_data.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: hist_lok_bndry_input_data.F,v 1.5 2005/06/10 13:29:08 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/hist_lok_bndry_input_data.F,v $
C   $Name: rel-5-5 $
C
C   SUBROUTINE RETRIEVES HISTORICAL BOUNDARY FLOW DATA AND MODIFIED
C   DELTA STORAGE FOR LAKE OKEECHOBEE
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE hist_lok_bndry_input_data (imth, ibm, ifyr, iyear,
     & istr_index, kflo0, mod_delta_stor, s65ernff, tcnsrnff, fplresws,
     & hist_lok_bndry_flw, dmdsemind, lp, nday_yr, numd,
     & nflpts, ifltabr, uisrnff)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'lakwca.inc'

      DIMENSION dmdsemind(366),fplresws(366),hist_lok_bndry_flw(11,366),
     & ifltabr(1200), istr_index(max_n_str_node_dep),
     & lp(60),numd(12),s65ernff(366),tcnsrnff(366),tmpvals(366),
     & kflo0(max_n_struc_flw,366),mod_delta_stor(366),uisrnff(366)

      CHARACTER  ca*5, cc*4, ce*4, cf*20,cpath*80, ctype*8, cunits*3,
     & sdate*9, stime*4, name_bndry_flw(11)*40

      LOGICAL lp

      REAL kflo0,mod_delta_stor
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

       ca = 'SFWMM'
       ce = '1DAY'
       cc = 'FLOW'
       stime = '1200'
       iyr = ifyr+iyear-1
c
c  write header to tape62 - file which contains capacity information
c    for key structures conveying water supply releases from LOK/WCA
c    to LEC Service areas
c
        if (lp(23)) then
         write(62,155) iyear,ifyr+iyear-1
 155     format(2(/),10X,'YEAR OF SIMULATION',i5,5x,'RAINFALL YEAR'
     $    ,i10)
        endif
      if (IMTH.EQ.IBM) then
         nday_yr = iymdjl(ifyr,ibm,1) - iymdjl(ifyr,1,1)
       else
         nday_yr = 0
       endif
c
c  read input from flo.dss file
c
c note: will only work if simulation starts on a january
c read one year's worth of data at the beginning of the year
c calculate number of days to read for this year
            nperiods = iymdjl(iyr,12,31) - iymdjl(iyr,1,1) + 1
c convert year,month,day integers to character format
            call ymddat(iyr,1,1,104,sdate,nd,ierror)
c do one dss read per structure ( one read = 365 or 366 daily values)
            do i=1,nflpts
             is = istr_index(i)
c form pathname
         if (flnm(is) .eq. 'G155  ' .or. flnm(is) .eq. 'G136  ' .or.
     &       flnm(is) .eq. 'G88   ' .or. flnm(is) .eq. 'G89   '
     &       .or. flnm(is) .eq. 'G155SP') then
            cf = 'BASE'
         elseif(flnm(is) .eq. 'S190  ') then
            if (isem_flg .eq. 1) then
             cf = 'EST_HIST_V5.0_REV'
            else
             cf = 'EST_HIST_V5.0'
            endif
         else
           cf = 'HISTORICAL'
         endif
               call zpath(ca,flnm(is),cc,sdate,ce,cf,cpath,npath)
c read from dss file
               call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &         tmpvals,cunits,ctype,iofset,istat)
               if(istat.ne.0)then
                 write(*,*) 'Error reading structure flow for ',flnm(is)
                 write(*,*) 'ISTAT:',istat
                 write(*,*) 'Consult HECDSS manual!!'
                 stop
```

```
               endif
c convert read values to integer values
               do j = 1,nperiods
                  kflo0(is,j) = tmpvals(j)
                  if (kflo0(is,j).eq.-901. .or. kflo0(is,j).eq.-902.)
     &             kflo0(is,j) = 0.0
               enddo
            enddo
c
c read Modified Delta Storage for LOK in flo.dss
c
        call zpath(ca,'MDS','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             mod_delta_stor(l) = tmpvals(l)
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for MDS'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c
c read S-65E from flo.dss
c
c
C  read in estimated flows thru S-65E into Lake Okeechobee
c
       if (typerun .ne. 'FUTURE ') then
        call zpath(ca,'S65E','FLOW',sdate,ce,
     &                    'HISTORICAL',cpath,npath)
       else
        call zpath(ca,'S65EFR','FLOW',sdate,ce,
     &                    'FUT_REV',cpath,npath)
       endif
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             s65ernff(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for S65E'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c
c read Taylor Creek/Nubbin Slough Flows from flo.dss
c
        call zpath(ca,'TCNSQ','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
```

```
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             tcnsrnff(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for TCNSQ'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c
c  read in estimated flows from Upper Istokpoga into Lake Okeechobee
c    (from S70+S75 via S71+S72)
c
        call zpath(ca,'UISTLK','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             uisrnff(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for UISTLK'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
cc define number of runoff and demand points input
c convert year,month,day integers to character format
      call ymddat(iyr,1,1,104,sdate,nd,ierror)

c  Define Historical LOK Boundary Condition Names
c
        name_bndry_flw(1)='NLSDM_REV'
        name_bndry_flw(2)='NLSRO_REV'
        name_bndry_flw(3)='NELSDMD'
        name_bndry_flw(4)='NELSRO'
        name_bndry_flw(5)='S4DMD'
        name_bndry_flw(6)='S4RO'
        name_bndry_flw(7)='ISTKABDMD'
        name_bndry_flw(8)='ISTKABRO'
        name_bndry_flw(9)='ISTKBBDMD'
        name_bndry_flw(10)='ISTKBBRO'
        name_bndry_flw(11)='S236DMD'

c  Extract Historical LOK Boundary Conditions from flo.dss
c
       do i = 1,11
         call zpath(ca,name_bndry_flw(i),'FLOW',sdate,ce,
     &                      'EST_HIST_V5.0',cpath,npath)
c        read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                  tmpvals,cunits,ctype,iofset,istat)
          do l = 1,nperiods
            hist_lok_bndry_flw(i,l) = tmpvals(l)*1.9835
          enddo
```

```
          if(istat.ne.0)then
             write(*,*) 'Error reading flo.dss flow for '
     &       ,name_bndry_flw(i)
             write(*,*) 'ISTAT:',istat
             write(*,*) 'Consult HECDSS manual!!'
             stop
          endif
       enddo
c
c  read in Flows to FPL Reservoir from LOK
c
       call zpath(ca,'FPLRES','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
       call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
        do l = 1,nperiods
          fplresws(l) = tmpvals(l)*1.9835
        enddo
        if(istat.ne.0)then
           write(*,*) 'Error reading flo.dss flow for FPLRES'
           write(*,*) 'ISTAT:',istat
           write(*,*) 'Consult HECDSS manual!!'
           stop
        endif
c
c read in estimated demands of Seminole indians on north side of
c Lake Okeechobee
c
        call zpath(ca,'SEMDMD','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
        call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             dmdsemind(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SEMIRR'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
C-------------------------------------------------------
C     READ ONE YEAR OF MONTHLY ET AND RAINFALL DATA FOR
C     LAKE OKEECHOBEE
C-------------------------------------------------------
      DO  KMONTH = 1,12
        READ(36,'(6X,5F10.0)') MOREFET(KMONTH), LOKMORN(KMONTH)
      ENDDO

      return
      end
```

holeyland_inflow_managmt.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: holeyland_inflow_managmt.F,v 1.9 2005/06/10 13:29:08 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/holeyland_inflow_managmt.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------
C   This subroutine calculates inflow into Holeyland based upon
c    options for management of the inflow (Rain-driven operations,
C    calendar-based operations,source of water).
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE holeyland_inflow_managmt (numd, water_supply_to_holyl,
     & rsiap, holyinq, totrun, pfrunf, trig_dpth_to_wt, qhlws)

      INCLUDE  'wmm_par.inc'
      INCLUDE  'abc.inc'
      INCLUDE  'agdata.inc'
      INCLUDE  'c1.inc'
      INCLUDE  'c2.inc'
      INCLUDE  'routc.inc'
      INCLUDE  'stat.inc'
      INCLUDE  'wrout.inc'
      INCLUDE  'stas.inc'
      INCLUDE  'simflows.inc'
      INCLUDE  'wcaot.inc'
      INCLUDE  'lakwca.inc'
      INCLUDE  'environ.inc'
      INCLUDE  'eaa_caps.inc'
      INCLUDE  'eaa_pump_caps.inc'
      INCLUDE  'wcasim.inc'
      INCLUDE  'opersched.inc'
      INCLUDE  'add_misc.inc'

      DIMENSION numd(12), pcapws(max_n_eaa_condt),
     & pfrunf(max_n_eaa_condt), pumpcap(max_n_eaa_condt),
```

```
     & qhlws(max_n_eaa_condt), wsred(max_n_eaa_condt)

      DOUBLE PRECISION holy_import_stage,hlstage
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (day1) THEN
        holpump=.false.
      END IF

      rmndp=0.5
      volhl=0.0
      volwsr=0.0
      qwsag=0.0
      qrnag=0.0
      wscap=0.0
      totcap=0.0
      nodews=0
      holyinq=0.0
      water_supply_to_holyl=0.0
C
C   SUPPLY FROM LAKE OKECHOBEE TO STAs. IF STAGE IN
C   LAKE BELOW THIS LEVEL DO NOT RELEASE WATER TO STAs
C----------------------------------------
      IF (month.eq.1) THEN
        prev=rsiapm(12)
      ELSE
        prev=rsiapm(month-1)
      END IF
      rsiap=prev+iday*(rsiapm(month)-prev)/numd(month)
C
C
C  DETERMINE VOLUME AVAILABLE FOR  WATER
C  SUPPLY DELIVERIES FROM THE HOLEY LAND RESERVOIR
C
      DO 10 iy=maxy,42,-1
        DO 10 node=isum(iy)+1,isum(iy+1)
        IF (ibsn(node).eq.96) THEN
          IF (pond(node).gt.rmndp) THEN
            volwsr=volwsr+(pond(node)-rmndp)*gdar
     &      *(1.0-carfact(node))
            nodews=nodews+1
          END IF
        END IF
   10 CONTINUE
C  ----------------------------------------------------
C  DETERMINE RUNOFF INTO RESERVOIR THAT WOULD BE PUMPED BY HLS1
C  ---------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      gwdpth=(ells(nodehl)-h(nodehl))*s(nodehl)
      IF (pond(nodehl).gt.gwdpth) THEN
        hlstage=pond(nodehl)-gwdpth+ells(nodehl)
      ELSE
        hlstage=(pond(nodehl)-gwdpth)/s(nodehl)+ells(nodehl)
      END IF
C
      CALL bpts (iday, month, 10.0, 7, 1, 1, holey_sched)
      CALL bpts (iday, month, 10.0, 7, 2, 1, holey_import_stg_ws_lok)
```

```
C
C  calculate target stage for holeyland
C
      IF (hlyenv.eq.'TRUE ') THEN
        avg_sim_wca_stg(iholy)=0.0
        DO i=1,ngrid_cells_daily(iholy)
          depthgw=(ells(node_import_daily(iholy,i))-
     &     h(node_import_daily(iholy,i)))*s(node_import_daily(iholy,i))
          IF (pond(node_import_daily(iholy,i)).gt.depthgw) THEN
            wca_nodal_stage=pond(node_import_daily(iholy,i))-depthgw+
     &       ells(node_import_daily(iholy,i))
          ELSE
            wca_nodal_stage=(pond(node_import_daily(iholy,i))-depthgw)/
     &       s(node_import_daily(iholy,i))+ells(node_import_daily(iholy,
     &       i))
          END IF
          avg_sim_wca_stg(iholy)=avg_sim_wca_stg(iholy)+wca_nodal_stage/
     &     ngrid_cells_daily(iholy)
        END DO
        hlstage=avg_sim_wca_stg(iholy)
        IF (wca_import_variation.eq.'DAILY') THEN
          stage_import_wca(iholy)=0.0
          DO i=1,ngrid_cells_daily(iholy)
            stage_import_wca(iholy)=stage_import_wca(iholy)+
     &       daily_stg_import_wca(ngage_index_env(iholy,i,1))/
     &       ngrid_cells_daily(iholy)
          END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
        ELSE
          IF (wca_import_variation.eq.'CONST') THEN
            stage_import_wca(iholy)=avg_import_stage(iholy)
          END IF
        END IF
        holy_import_stage=stage_import_wca(iholy)
      ELSE
        holy_import_stage=holey_sched
      END IF
C
      IF (hlstage.lt.holy_import_stage.or.holpump) THEN
        holpump=.true.
        holpump=.not.(hlstage.ge.(holy_import_stage+0.2))
        IF (holpump) THEN
          IF (hlstage.le.holy_import_stage) THEN
            DO 20 ia=1,neaabsn-1
              pumpcap(ia)=pcfs(ia)*86400.
   20       CONTINUE
          ELSE
            DO 30 ia=1,neaabsn-1
              pumpcap(ia)=pcfs(ia)/0.2*(holy_import_stage+0.2-hlstage)*
     &         86400.
   30       CONTINUE
          END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
        ELSE
          DO 40 ia=1,neaabsn-1
            pumpcap(ia)=0.0
   40     CONTINUE
```

```
        END IF
      ELSE
        DO 50 ia=1,neaabsn-1
          pumpcap(ia)=0.0
   50   CONTINUE
      END IF
C
C   DELIVER WATER TO HOLEY LAND TO PREVENT BRUSH FIRES
C
      IF (ws_to_holy_opt.eq.'BRUSHF') THEN
        nfire=0
        IF (hlyenv.eq.'TRUE ') THEN
          trig_dpth_to_wt=max(ells(nodehl)-holy_import_stage+
     &     offset(iholy,1)+0.25,0.0)
          trig_dpth_to_wt=max(trig_dpth_to_wt,1.0)
        ELSE
          trig_dpth_to_wt=max(ells(nodehl)-holey_import_stg_ws_lok,
     &     0.0)
        END IF
        IF (ells(nodehl)-h(nodehl)
     &   .gt.trig_dpth_to_wt.and.holey_min_level_opt.eq.'YES') THEN
          DO 60 iy=maxy,42,-1
            DO 60 node=isum(iy)+1,isum(iy+1)
            IF (ibsn(node).eq.96) THEN
              IF (ells(node)-h(node).gt.trig_dpth_to_wt) THEN
                dphgw=(ells(node)-h(node)-trig_dpth_to_wt)*s(node)
                IF (pond(node).le.dphgw) THEN
                  sdiff=dphgw-pond(node)
                  volhl=volhl+sdiff*gdar
                  nfire=nfire+1
                END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              END IF
            END IF
   60     CONTINUE
        END IF
      END IF
C -------------------------------------------------------
C    DETERMINE TOTAL RUNOFF AND/OR CROP DEMAND FROM EAA
C -------------------------------------------------------
C
      DO 70 ia=1,neaabsn-1
        pcapws(ia)=pcfws(ia)*86400.
        IF (nsindx(ia).eq.1) THEN
          IF (agp(ia).ge.0.0) THEN
            qwsag=qwsag+agp(ia)
            wscap=wscap+min(agp(ia),pcapws(ia))
            IF (agp(ia).gt.0) THEN
              wsred(ia)=min(agp(ia),pcapws(ia))/agp(ia)
            ELSE
              wsred(ia)=0.0
            END IF
            flrnro(ia)=0.0
          ELSE
            IF (runoff_to_holeyland.eq.'DIRECT ') THEN
              IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
                agrnff=pfrunf(ia)*totrun*86400.*bmpred
```

```
              ELSE
                agrnff=abs(agp(ia))*bmpred
              END IF
              agrnff=agrnff-amin1(flow_reduction_bmp(ia)*86400.,agrnff)
C
              qrnag=qrnag+agrnff
              totcap=totcap+amin1(agrnff,pumpcap(ia))
              IF (agrnff.gt.0.0) THEN
                flrnro(ia)=amin1(agrnff,pumpcap(ia))/agrnff
              ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                flrnro(ia)=0.0
              END IF
              qu(int_cnl_holey(ia))=qu(int_cnl_holey(ia))+flrnro(ia)*
     &         agrnff
              holyinq=holyinq+flrnro(ia)*agrnff
            ELSE
              flrnro(ia)=0.0
            END IF
          END IF
        ELSE
          flrnro(ia)=0.0
        END IF
   70 CONTINUE
C
      qrnoffm=qrnoffm+totcap/43560.
C
      IF (ells(nodehl)-h(nodehl)
     &.gt.trig_dpth_to_wt.and.stagelo.gt.rsiap.and.ws_to_holy_opt.eq.'BR
     &USHF'.and.holey_min_level_opt.eq.'YES') THEN
        DO 80 ia=1,neaabsn-1
          IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
            agrnff=pfrunf(ia)*totrun*bmpred*86400.
          ELSE
            IF (agp(ia).lt.0.0) THEN
              agrnff=abs(agp(ia))*bmpred
            ELSE
              agrnff=0.0
            END IF
          END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          IF (runoff_to_holeyland.eq.'INDIRCT') THEN
            runoff_to_holyl=0.0
          ELSE
            runoff_to_holyl=agrnff
          END IF
C
          ptotcp=pumpcap(ia)-min(runoff_to_holyl,pumpcap(ia))
          volhl=volhl-flrnro(ia)*agrnff
          volhl=amax1(volhl,0.0)
          qhlws(ia)=min(ptotcp,volhl)
          volhl=volhl-qhlws(ia)
          pumpcap(ia)=ptotcp-qhlws(ia)
   80   CONTINUE
      ELSE
        DO 90 ia=1,neaabsn-1
```

```
          qhlws(ia)=0.0
   90   CONTINUE
      END IF
C
      IF (runoff_to_holeyland.eq.'DIRECT') THEN
        DO ia=1,neaabsn-1
          IF (ires_pump(ia).ne.-901) THEN
            pumpcap_res(ires_pump(ia))=0.0
          END IF
        END DO
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
        DO ia=1,neaabsn-1
          IF (ires_pump(ia).ne.-901) THEN
            pumpcap_res(ires_pump(ia))=pumpcap(ia)/86400.+
     &       pumpcap_res(ires_pump(ia))
          END IF
        END DO
      ELSE
        DO ip=1,npumptores
          IF (ibsnres(ip).eq.96) THEN
            pumpcap_res(ip)=pumpcap(ip)/86400.
          END IF
        END DO
      END IF
C
C  CALCULATE NET VOLUME OF FLOW INTO HOLEY LAND RESERVOIR
C
      IF (qwsag.ge.0.0) THEN
        IF (hlstage-ells(nodehl).gt.rmndp) THEN
          flwtrs=min(wscap,volwsr)
          IF (wscap.gt.0.0) THEN
            wsratio=flwtrs/wscap
          ELSE
            wsratio=0.0
          END IF
        ELSE
          flwtrs=0.0
          wsratio=0.0
        END IF
        DO 100 ia=1,neaabsn-1
          IF (nsindx(ia).eq.1) THEN
            flwsro(ia)=wsred(ia)*wsratio
          ELSE
            flwsro(ia)=0.0
          END IF
  100   CONTINUE
      END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (nodews.gt.0) THEN
        DO 110 iy=maxy,42,-1
          DO 110 node=isum(iy)+1,isum(iy+1)
          IF (ibsn(node).eq.96) THEN
            IF (pond(node).gt.rmndp) THEN
              pond(node)=pond(node)-abs(flwtrs)/(gdar*nodews
     &        *(1.0-carfact(node)))
```

```
            END IF
          END IF
  110   CONTINUE
        qwtspm=qwtspm+abs(flwtrs)/43560.
      END IF
C
      RETURN
      END
```

init.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: init.F,v 1.25 2005/06/10 13:29:09 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/init.F,v $
C   $Name: rel-5-5 $
C
C
C   PURPOSE: INITIALIZE APPROPRIATE VARIABLES IN THE MODEL. THESE
C   VARIABLES NEED TO BE INITIALIZED ONCE.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE init (volwtr, rstgpls, rlist_lvname, import_area_name,
     & ntstrctr, numd, nyday, runtitle, wsuply, tot_cnl_dmd_asr,
     & outflow_to_canal_roten, outflow_to_marsh,
     & flow_to_sta3_fr_298_nnrc, flow_to_sta2_fr_298d, stagelo_res,
     & lok_split_option, nmdays_lok, climate_index, no_of_hurr_pred,
     & tot_runoff_excess, hres01, asr_vol01, hres02, asr_vol02, itypern,
     & iwsbpcntr, nca_r, s356_to_cell, s356_to_l29, s333_reg_lec,
     & s337fc, s334_fc, iroute_to_wca3a_frm_holy, total_lec_dmnd_lok,
     & tot_avl_l8_bpit_res_m1q,sum_eaa_dmnd_ssm_nc,prevqu,
     & c304_out_to_marsh,sum_eaa_dmnd_ssm,sum_alocnnr,ws_from_lo_s6,
     & rnlkblt_to_s29,dellec,sum_alloc_ssm,sum_losa_dmnd,
     & igravity_s8_nosta,igravity_s7_nosta,tempf_weircff,temp_weircff,
     & flow_to_c9r_frm_c11,ws_frm_lok_to_semcyp,
     & dischg_to_bay_s29_frm_c11r,rloc_excss_to_canl_next_step,
     & istokabdmdnotmet,pctmet,flow_to_sta1w_fr_298d,
     & out_flow_east,out_flow_east0)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
```

```
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'loktarg.inc'
      INCLUDE 'resbudg.inc'

      DIMENSION climate_index(12), numd(12), rstgpls(max_n_puls_rel),
     & tot_runoff_excess(max_n_eaa_condt), volwtr(max_n_res),
     & wsuply(max_n_res), sum_eaa_dmnd_ssm_nc(max_n_eaa_condt),
     & prevqu(max_ncnls),sum_eaa_dmnd_ssm(max_n_eaa_condt),
     & tempf_weircff(max_n_res,max_nostr_res),
     & temp_weircff(max_n_res,max_nostr_res),
     & rloc_excss_to_canl_next_step(max_ncnls)

      CHARACTER import_area_name(max_n_targ_area)*5,
     & rlist_lvname(max_no_lvsp)*5, runtitle*256

Cwmw To be used by get tide
      INTEGER midmonth(12)
      REAL mainky(12), flamingo(12), virgkey(12), hollywd(12),
     & delray(12), pbeach(12), stuart(12), istokabdmdnotmet

      DATA midmonth /15,45,74,105,135,166, 196,227,258,288,319,349/
C   the foll are the 12 midmonth long time avg mean sea level values for the 6
stations(ft NGVD)
      DATA flamingo /-0.06,-0.05,0.03,0.13,0.17,0.21,0.235,0.35,
     & 0.47,0.46,0.255,0.055/
      DATA mainky /0.47,0.48,0.49,0.52,0.52,0.53,0.53,0.56,
     & 0.58,0.58,0.54,0.50/
      DATA virgkey /0.57,0.70,0.58,0.65,0.67,0.68,0.62,0.78,
     & 1.10,1.15,0.87,0.61/
      DATA hollywd /0.59,0.72,0.60,0.67,0.69,0.69,0.63,
     & 0.80,1.12,1.18,0.89,0.63/
      DATA delray /0.31,0.47,0.32,0.41,0.44,0.44,0.37,0.58,
     & 0.96,1.04,0.68,0.36/
      DATA pbeach /0.17,0.34,0.18,0.27,0.31,0.31,0.23,0.46,0.88,
     & 0.96,0.57,0.22/
      DATA stuart /0.98,1.04,0.99,1.02,1.03,1.03,1.00,1.08,1.24,
     & 1.26,1.13,1.00/
Cwmw endcomm

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      NODEF(IX,IY) = IX - MINX(IY) + 1 + ISUM(IY)
      NZERO = 0
      IFLS339 = 40
      IFLS340 = 41
      igravity_s8_nosta = 1
      igravity_s7_nosta = 1

C  INITIALIZE VOLUME OF WATER STAS CAN RECEIVE
```

```
C  REMAINS ZERO THROUGOUT SIMULATION IF NO STAS EXIST

      DO 10 I = 1,max_n_res
         VOLWTR(I) = 99999.
       DO 9 KK = 1,max_nostr_res
         tempf_weircff(I,KK) = 99999.
         temp_weircff(I,KK) = 99999.
    9  CONTINUE
   10 CONTINUE

C  PERFORM LOOP IF HAVE PROPOSED STAS in SIMULATION

      IF (NTOTRES .GT. 0) THEN
         DO 60 IA = 1,NTOTRES
            DO 30 IS = 1,NINSTR(IA)
               IF (INODOPT(IA,IS,1) .EQ. 1) THEN
                  DO 20 J = 1,NINDPTS(IA,IS)
                     IUPSTRM(IA,IS,J) = NODEF(IXLIN(IA,IS,J),IYLIN
     $               (IA,IS,J))
   20             CONTINUE
               ENDIF
   30       CONTINUE
            DO 50 J = 1,NOUTSTR(IA)
               IF (INODOPT(IA,J,2) .EQ. 1) THEN
                  IDNSTRM(IA,J) = NODEF(IXLOUT(IA,J),IYLOUT(IA,J))
               ENDIF
               IF (INODOPT(IA,J,3) .EQ. 1) THEN
                  DO 40 K = 1,NDSPTS(IA,J)
                     NODEPW(IA,J,K) = NODEF(IXRW(IA,J,K),IYRW(IA,J,K))
   40             CONTINUE
               ENDIF
   50       CONTINUE
            NODEWA(IA) = NODEF(IXCNODE(IA),IYCNODE(IA))
   60    CONTINUE
C
      ENDIF
C
C  INITIALIZE REFERENCE LAKE STAGE ARRAY FOR PULSE
C  RELEASES FROM LAKE OKEECHOBEE TO ST LUCIE CANAL
C  AND CALOOSAHATCHEE RIVER
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      DO 80 I = 1,10
         rstgpls(I) = 0.0
   80 CONTINUE
      IPULSE = 0
      iroute_to_wca3a_frm_holy = 0
      total_lec_dmnd_lok = 0.0
      istokabdmdnotmet = 0.0
      sum_alocnnr = 0.0
      out_flow_east = 0.0
      out_flow_east0 = 0.0
      ws_from_lo_s6 = 0.0
      rnlkblt_to_s29 = 0.0
      flow_to_c9r_frm_c11 = 0.0
      ws_frm_lok_to_semcyp = 0.0
      dischg_to_bay_s29_frm_c11r = 0.0
```

```
      dellec = 0.0
      pctmet = 1.0
      sum_alloc_ssm = 0.0
      sum_losa_dmnd = 0.0
      tot_avl_l8_bpit_res_m1q = 0.0
      do i = 1,max_ncnls
         rloc_excss_to_canl_next_step(i) = 0.0
      enddo
      do i=1,max_n_eaa_condt
         sum_eaa_dmnd_ssm_nc(i) = 0.0
         sum_eaa_dmnd_ssm(i) = 0.0
      enddo
      c304_out_to_marsh = 0.0
      sspsfcd_to_lok = 0.0
      sspsfcd_to_sta = 0.0
      eswcd_to_lok  = 0.0
      eswcd_to_sta  = 0.0
      q715_farms_to_lok = 0.0
      q715_farms_to_sta = 0.0
      ebwcd_to_lok = 0.0
      ebwcd_to_sta = 0.0
      NCA = 0
      NCA_OV = 0
      NCA_ALT = 0
      NCA_R = 0
      DAY1 = .TRUE.
      DO n=1,nch
       DO l=1,nch
         prev_outf_to_ds_cnl(n,l) = 0.0
       ENDDO
      ENDDO
C
      IG31W   = NODEF(23,10)
      NODS140 = NODEF(14,39)
      NODCAT  = NODEF(37,58)
      LOXNOD1  = NODEF(36,59)
      LOXNOD2  = NODEF(36,60)
      NODNWC3A = NODEF(17,41)
      IGS357DN = NODEF(24,17)
      NOD_DS_S11AB = NODEF(27,36)
      NOD_DS_S11C  = NODEF(27,38)
      NOD_DS_S9 = NODEF(27,32)
      NOD_DS_S150 = NODEF(25,41)
      IROTDNOD1 = NODEF(16,41)
      IROTDNOD2 = NODEF(17,41)
      IG596 = NODEF(26,18)
      NODE_G3273 = NODEF(24,17)
      NODES9 = NODEF(28,32)
      IG3B29 = NODEF(26,26)
      IG3B2 = NODEF(25,27)
      IG3B3 = NODEF(24,26)
      IG3B4 = NODEF(24,23)
      IG2517 = NODEF(25,17)
      NODL101 = NODEF(30,53)
      ista1e_node = nodef(31,53)
      NODG261 = NODEF(25,62)
      NODG262 = NODEF(26,61)
```

```
      NODG263 = NODEF(27,60)
      NODL8R2 = NODEF(28,59)
      NODL8R1 = NODEF(27,60)
      node_dn_s355 = NODEF(24,22)
      IG3A4  = NODEF(21,29)
      IG3A2 = NODEF(19,36)
      IG618 = NODEF(23,22)
      IGL38 = NODEF(30,38)
      IG3A9 = NODEF(22,34)
      IG3A3 = NODEF(25,37)
      IG3A28 = NODEF(19,24)
      IG2431 = NODEF(24,31)
      NODANGL = NODEF(26,17)
c
c  structure indices
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      CALL match (strname_spec(51),1,6,flnm,300,6,ifls333)
      if (ifls333 .eq. 0) ifls333 = NTSTRCTR + 1
      CALL match (strname_spec(45),1,6,flnm,300,6,ifls12)
      IF (ifls12 .EQ. 0) THEN
         CALL match(strname_spec(49),1,6,flnm,300,6,ifls12)
      ENDIF
      CALL match ('S12A  ',1,6,flnm,300,6,ifls12a)
      CALL match ('S12B  ',1,6,flnm,300,6,ifls12b)
      CALL match ('S12C  ',1,6,flnm,300,6,ifls12c)
      CALL match ('S12D  ',1,6,flnm,300,6,ifls12d)
      CALL match (struc_name_spec_code(9),1,6,flnm,300,6,ifls150)
      CALL match (name_struc_w_spec_code_wca(9),1,6,flnm,300,6,ifls11)
      CALL match (name_struc_w_spec_code_wca(10),1,6,flnm,300,6,ifls144)
      CALL match (name_struc_w_spec_code_wca(11),1,6,flnm,300,6,ifls145)
      CALL match (name_struc_w_spec_code_wca(12),1,6,flnm,300,6,ifls146)
      CALL match (name_struc_w_spec_code_wca(7),1,6,flnm,300,6,ifls39)
      CALL match (struc_name_spec_code(14),1,6,flnm,300,6,ifls357)
      CALL match (strname_spec(6),1,6,flnm,300,6,ifls335)
      CALL match (strname_spec(4),1,6,flnm,300,6,ifls336)
      CALL match (strname_spec(64),1,6,flnm,300,6,ifls174)
      CALL match ('S332  ',1,6,flnm,300,6,ifls332)
      CALL match (strname_spec(5),1,6,flnm,300,6,ifls338)
      CALL match (strname_spec(7),1,6,flnm,300,6,ifls337)
      CALL match (strname_spec(31),1,6,flnm,300,6,ifls356)
      CALL match (strname_spec(68),1,6,flnm,300,6,ifls197)
      CALL match (canl_names_spec(44),1,5,cnm,nch+1,5,ista34)
      CALL match (name_struc_w_spec_code_wca(13),1,6,flnm,300,6,ifls38)
      IF (runmode .EQ. 'CALIB') THEN
         CALL match ('S151  ',1,6,flnm,300,6,ifls151)
      ELSE
         CALL match (name_struc_w_spec_code_wca(18),1,6,flnm,300,6
     $    ,ifls151)
      ENDIF
      IF (runmode .EQ. 'CALIB') THEN
         CALL match ('S31   ',1,6,flnm,300,6,ifls31)
      ELSE
         CALL match(name_struc_w_spec_code_wca(19),1,6,flnm,300,6,
     $   ifls31)
      ENDIF
      CALL match (name_struc_w_spec_code_wca(15),1,6,flnm,300,6,ifls34)
```

```
      CALL match (strname_spec(9),1,6,flnm,300,6,ifls155)
      CALL match ('G89   ',1,6,flnm,300,6,iflg89)
      CALL match ('S4    ',1,6,flnm,300,6,ifls4)
      CALL match ('S235  ',1,6,flnm,300,6,ifls235)
      CALL match ('S65E  ',1,6,flnm,300,6,ifls65e)
      ILAGBSN = 10
      IL8AGBSN = 40
      CALL match (strname_spec(75),1,6,flnm,300,6,ifls334)
      CALL match (strname_spec(50),1,6,flnm,300,6,ifls355)
      CALL match (name_struc_w_spec_code_wca(29),1,6,flnm,300,6,ifls345)
      CALL match (struc_name_spec_code(12),1,6,flnm,300,6,iflagq)
      CALL match ('M_CNL ',1,6,flnm,300,6,ifls2mc)
      CALL match (name_struc_w_spec_code_wca(14),1,6,flnm,300,6,ifls142)
      CALL match ('S141  ',1,6,flnm,300,6,ifls141)
      CALL match (name_struc_w_spec_code_wca(8),1,6,flnm,300,6,ifls143)
      CALL match (name_struc_w_spec_code_wca(2),1,6,flnm,300,6,iflpsac)
      CALL match (struc_name_spec_code(20),1,6,flnm,300,6,iflc10a)
      CALL match ('G136  ',1,6,flnm,300,6,ig136)
      CALL match (strname_spec(83),1,6,flnm,300,6,iflc11r)
      CALL match (strname_spec(84),1,6,flnm,300,6,iflc9r)
      CALL match ('S349A ',1,6,flnm,300,6,ifls349a)
      CALL match ('S349B ',1,6,flnm,300,6,ifls349b)
      if (ifls349a .eq. 0) ifls349a=NTSTRCTR + 1
      if (ifls349b .eq .0) ifls349b=NTSTRCTR + 1
      IF (iflc11r .EQ. 0) iflc11r = NTSTRCTR + 1
      IF (iflc9r  .EQ. 0) iflc9r  = NTSTRCTR + 1
      IWPBCAT = NODEF(36,57)
      CALL match (strname_spec(82),1,6,flnm,300,6,iflg124)
c
c canal indices
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      CALL match (canl_names_spec(40),1,5,cnm,nch+1,5,innrc)
      CALL match (canl_names_spec(4),1,5,cnm,nch+1,5,ic13)
      CALL match (canl_names_spec(6),1,5,cnm,nch+1,5,ic12)
      CALL match (canl_names_spec(28),1,5,cnm,nch+1,5,l31s)
      CALL match (canl_names_spec(27),1,5,cnm,nch+1,5,l31nc)
      CALL match (canl_names_spec(34),1,5,cnm,nch+1,5,l31w)
      CALL match (canl_names_spec(17),1,5,cnm,nch+1,5,lc111e)
      CALL match (canl_names_spec(16),1,5,cnm,nch+1,5,lc111)
      CALL match (canl_names_spec(23),1,5,cnm,nch+1,5,lhls)
      CALL match (canl_names_spec(5),1,5,cnm,nch+1,5,is37b)
      CALL match (canl_names_spec(41),1,5,cnm,nch+1,5,iroten_canal_no)
      CALL match (canl_names_spec(48),1,5,cnm,nch+1,5,ldculv1)
      CALL match (canl_names_spec(20),1,5,cnm,nch+1,5,ldculv2)
      IF (iroten_canal_no .EQ. 0) iroten_canal_no = nch+1
      CALL match (canl_names_spec(18),1,5,cnm,nch+1,5,ic304)
      CALL match (canl_names_spec(29),1,5,cnm,nch+1,5,il33)
      CALL match (canl_names_spec(30),1,5,cnm,nch+1,5,il37)
      CALL match (canl_names_spec(38),1,5,cnm,nch+1,5,il38)
      CALL match (canl_names_spec(33),1,5,cnm,nch+1,5,il38e)
      CALL match (canl_names_spec(14),1,5,cnm,nch+1,5,ic11w)
      CALL match (canl_names_spec(13),1,5,cnm,nch+1,5,ic11)
      CALL match (canl_names_spec(36),1,5,cnm,nch+1,5,is140up)
      CALL match (canl_names_spec(37),1,5,cnm,nch+1,5,is339up)
      CALL match (canl_names_spec(1),1,5,cnm,nch+1,5,iacmdr)
      CALL match (canl_names_spec(2),1,5,cnm,nch+1,5,ic18)
```

```
c
      CALL match (canl_names_spec(7),1,5,cnm,nch+1,5,ic4)
      CALL match (canl_names_spec(8),1,5,cnm,nch+1,5,imatch)
      CALL match (canl_names_spec(43),1,5,cnm,nch+1,5,isncre)
      CALL match (canl_names_spec(31),1,5,cnm,nch+1,5,l30)
      IF (imatch .NE. 0) ic4 = imatch
      CALL match (canl_names_spec(10),1,5,cnm,nch+1,5,is22up)
      CALL match (canl_names_spec(9),1,5,cnm,nch+1,5,is25bup)
      CALL match (canl_names_spec(35),1,5,cnm,nch+1,5,ilkmg)
      CALL match (canl_names_spec(11),1,5,cnm,nch+1,5,ic51w)
      IF (ic51w .EQ. 0) ic51w = nch+1
      CALL match ('L29  ',1,5,cnm,nch+1,5,l29)
      if (l29 .eq. 0) l29 = nch+1
      CALL match (canl_names_spec(39),1,5,cnm,nch+1,5,imcnl)
      IF (iconvey_canal_no(is150bws-6) .NE. 0) THEN
         ica3 = intcnl(idn_wca_indx(is150bws-6) ,iconvey_canal_no
     $   (is150bws-6))
      ELSE
         ica3 = nch + 1
      ENDIF
      IF (iconvey_canal_no(is6bsn-6) .NE. 0) THEN
         ica1 = intcnl(idn_wca_indx(is6bsn-6) ,iconvey_canal_no
     $   (is6bsn-6))
      ELSE
         ica1 = nch + 1
      ENDIF
      IF (opt_l8_prop) THEN
         CALL match (canl_names_spec(26),1,5,cnm,nch+1,5,il8s)
         CALL match (canl_names_spec(25),1,5,cnm,nch+1,5,il8w)
      ELSE
         il8s = nch+1
         il8w = nch+1
      ENDIF
      CALL match (canl_names_spec(20),1,5,cnm,nch+1,5,idculv1)
      CALL match (canl_names_spec(21),1,5,cnm,nch+1,5,idculv2)
      CALL match ('G155  ',1,6,flnm,300,6,ig155)
      CALL match ('G89   ',1,6,flnm,300,6,ig89)
      CALL match ('G88   ',1,6,flnm,300,6,ig88)
      CALL match ('G200A ',1,6,flnm,300,6,ig200a)
      CALL match (strname_spec(81),1,6,flnm,300,6,ifls176)
c  levees
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      CALL match ('L35   ',1,5,rlist_lvname,nlevs,5,ilv37)
      CALL match ('L33   ',1,5,rlist_lvname,nlevs,5,ilv33)
      IF (iresid .EQ. 0) iresid = nch+1
      IBC51W = 5
      iwca1 = idn_wca_indx(is6bsn-6)
      iwca2a = idn_wca_indx(is7bsn-6)
      iwca3a = idn_wca_indx(is8bsn-6)
      CALL match('WCA3B',1,5,wca_name,nwcas,5,iwca3b)
      IF (iwca3b .EQ. 0) iwca3b = iwca3a
      CALL match('WCA2B',1,5,wca_name,nwcas,5,iwca2b)
      IF (iwca2b .EQ. 0) iwca2b = idn_wca_indx(is7bsn-6)
      CALL match('NESRS',1,5,import_area_name,ntotal_no_import_areas ,5,
     $inesrs)
      CALL match('NWPRK',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iwenp)
```

```
      CALL match('HOLYL',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iholy)
      CALL match('ROTEN',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iroten)
      CALL match('BCNP ',1,5,import_area_name,ntotal_no_import_areas ,5,
     $ibcnp)
      IF (ibcnp .EQ. 0) ibcnp = ntotal_no_import_areas+1
      CALL match('LOXSL',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iloxah)
      CALL match('NWC3A',1,5,import_area_name,ntotal_no_import_areas ,5,
     $inwcwca3a)
      CALL match('W3A28',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iwca3a_3a28)
      CALL match('S10DS',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iwca2a_2a17)
      CALL match('WC3A3',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iwca3a_3a3)
      CALL match('WC3A2',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iwca3a_3a2)
      CALL match('WC3BE',1,5,import_area_name,ntotal_no_import_areas ,5,
     $iewca3b)
      IF (iewca3b .EQ. 0) iewca3b = ntotal_no_import_areas+1
      call match('WC3BC',1,5,import_area_name,ntotal_no_import_areas
     $      ,5,icwca3b)
      if (icwca3b .eq. 0) icwca3b = ntotal_no_import_areas+1
      CALL match('TSLGH',1,5,import_area_name,ntotal_no_import_areas ,5,
     $itylsl)
      CALL match('L31W ',1,5,import_area_name,ntotal_no_import_areas ,5,
     $il31w_min_rf)
      CALL match('WC3NE',1,5,import_area_name,ntotal_no_import_areas ,5,
     $inewca3a)
      CALL match('W3APK',1,5,import_area_name,ntotal_no_import_areas ,5,
     $ig3a28_enp)
      CALL match('S333F',1,5,import_area_name,ntotal_no_import_areas ,5,
     $igs333_enp)
      CALL match('G1502',1,5,import_area_name,ntotal_no_import_areas ,5,
     $ig1502)
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IMIAMI = 1
      INNR   = 2
      IHILL  = 4
      IWPB   = 3
C
      ISA1 = 1
      ISA2 = 2
      ISA3 = 3
C
      NODEHL = NODEF(20,43)
      NODE_ROTEN1 = NODEF(16,43)
      NODE_ROTEN2 = NODEF(15,46)
c
      DO i = 1,ntotres
         wsuply(i) = 0.0
         rinflw_to_res_frm_cnl(i) = 0.0
         res_struc_inflw(i) = 0.0
      END DO
```

```
c
      DO n = 1,nch
         ws_status_cnl(n) = 'MAINT'
         opt_fc_canal(n)  = 'FCLOPS'
         prevqu(n)=0.0
         do nstr=1,nstrctr(n)
           do i = 1,nsubstr(n,nstr)
            icount_bel_fc_thres(n,nstr,i) = 0
           enddo
         enddo
      END DO
      DO i = 1,no_flwth_cnls
         ws_status_cnl(icnl_flwth(i)) = 'FLWTH'
      END DO
      DO i = 1,no_fc_flwth_cnls
         opt_fc_canal(icnl_fc_flwth(i)) = 'FLWTH '
      END DO
      tot_avl_l8_bpit_res = 0.0
      tot_cnl_dmd_asr = 0.0
      flow_to_sta3_fr_298_nnrc = 0.0
      flow_to_sta2_fr_298d = 0.0
      flow_to_sta1w_fr_298d = 0.0
      total_outflow_from_roten = 0.0
      total_outf_from_holy_g200a = 0.0
      outflow_from_roten_to_west = 0.0
      outf_to_c9res_via_us27s = 0.0
      outflow_from_roten_to_wca = 0.0
      outflow_from_roten_to_holy = 0.0
      sta34_out_to_s140 = 0.0
      sta34_out_to_wca3ane = 0.0
      outflow_from_sta6_to_s140 = 0.0
      rem_inflow_to_meet_holy_sched = 0.0
      outflow_from_sta6 = 0.0
      outflow_to_canal_roten = 0.0
      outflow_to_marsh = 0.0
      outf_from_c11res = 0.0
      IF (lok_split_option .EQ. 0) stagelo_res = -999.0
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      nmdays = 0
      ndays_sim = 0
      ndays_sim_sd = 0
      nmdays_lok = 0
      DO i = 1,12
         climate_index(i) = 0.0
      END DO
      no_of_hurr_pred = 0
      DO i = 1,neaabsn
         tot_runoff_excess(i) = 0.0
      END DO
c
      DO node = 1,max_ncells
         outf_seep_loc_to_res_cell(node) = 0.0
         outf_fc_frm_res_to_cell(node) = 0.0
         seep_loss_res(node) = 0.0
         adjust_for_area(node) = 0.0
         adjust_for_area_n(node) = 0.0
```

```
         adjust_for_area_m(node) = 0.0
         adjust_for_area_m1(NODE) = 0.0
         adjust_for_area_m2(NODE) = 0.0
         adjust_for_area_m3(NODE) = 0.0
         adjust_for_area_m4(NODE) = 0.0
         sum_chg_dpth_ovl_flw(NODE) = 0.0
         sum_chg_depth_levee_spg(node) = 0.0
         chg_depth_levee_spg(node) = 0.0
         sum_chg_depth_levee_spg_unadj(node) = 0.0
         daily_excess_adjust(node) = 0.0
         seep_loss_res_cell(NODE) = 0.0
         pond_adj_due_to_res(node) = 0.0
         daily_outf_from_res_to_cell_max(node) = 0.0
         daily_outf_seep_loc_to_res_cell(node) = 0.0
         DO ir = 1,noresincell(NODE)
            prev_stage_res(node,ir) = stage_res(node,ir)
         ENDDO
      END DO
c
      DO i = 1,40
        avg_sim_wca_stg_out(i) = 0.0
      ENDDO
c
      IF (demand_level_opt .EQ. 'CURR') THEN
         itypern = 1
      ELSE
         itypern = 2
      ENDIF
C
      IF (MOD(IFYR,4) .EQ. 0) THEN
         NUMD(2) = 29
      ELSE
         NUMD(2) = 28
      ENDIF
C
      NYDAY = 0
      DO 90 IM = 1,IBM-1
         NYDAY = NYDAY + NUMD(IM)
   90 CONTINUE
C
      DO I = 1,5
        LFORPUMP(I) = .FALSE.
      ENDDO
      DO I=1,NWCAS
        LFRACBFLR(I) = .FALSE.
      ENDDO
      LS151 = .TRUE.
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      s333_reg_lec = 0.0
      s356_to_l29 = 0.0
      s356_to_cell = 0.0
      s337fc = 0.0
      s334_fc = 0.0
      outf_fc_thru_s381 = -901.
      hres01 = 0.0
      asr_vol01 = 0.0
```

```
      hres02 = 0.0
      asr_vol02 = 0.0
      IWSBPCNTR = 0
      nmdays_rainavg = 0
      nmdays_rainavg0 = 0
      IF (IBM .GT.  2 .AND. MOD(IFYR,4) .EQ. 0) NYDAY = NYDAY - 1
C
c  WRITE runtitle to tape62:summary of water supply deliveries
c  at major structures
c
      IF (lp(23)) THEN
         WRITE(62,'(A)') RUNTITLE
      ENDIF
C
C Data to be used by get_tide function
C
      DO i=1,366
         xval1=i
         yval(i) = tideinterp(xval1,midmonth,flamingo,12)
         xval1=i
         yval1(i) = tideinterp(xval1,midmonth,mainky,12)
         xval1=i
         yval2(i) = tideinterp(xval1,midmonth,virgkey,12)
         xval1=i
         yval3(i) = tideinterp(xval1,midmonth,hollywd,12)
         xval1=i
         yval4(i) = tideinterp(xval1,midmonth,delray,12)
         xval1=i
         yval5(i) = tideinterp(xval1,midmonth,pbeach,12)
         xval1=i
         yval6(i) = tideinterp(xval1,midmonth,stuart,12)
      END DO
      RETURN
      END
```

knflows.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: knflows.F,v 1.17 2005/06/10 13:29:09 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/knflows.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE DISTRIBUTES MEASURED OR SIMULATED (PRIOR TO CHNLF
C   SUBROUTINE) FLOWS TO THE APPROPRIATE CANAL OR SURFACE PONDING LOCATION.
C
C              THE IOPT PARAMETER SPECIFIES HOW THE FLOWS
C                     ARE DISTRIBUTED.
C
C        IOPT = 1          FLOW TO PONDING
C        IOPT = 2          FLOW TO A CANAL
C        IOPT = 3    FLOW FROM POND1 TO POND2
C        IOPT = 4    FLOW FROM POND TO CANAL
C        IOPT = 5    FLOW FROM CANAL TO POND
C        IOPT = 6    FLOW FROM CANAL TO CANAL
C        IOPT = 7    FLOW FROM EAA BASIN TO CANAL
C        IOPT = 8    FLOW FROM EAA BASIN TO GRID CELL
C
C   SEASONAL ADJUSTMENTS TO REGULATION SCHEDULES ARE ALSO
C            MADE IN  THIS SUBROUTINE
C   VARIABLES:
C      CFDP      CONV. FACTOR FROM CFS-DAYS TO DEPTH
C      CNWORD    CHARACTER VARIABLE FOR FLOW DISTRIBUTION PARA.
C      NFLO(3)   MONTHLY MIN, MAX, AND TOTAL FLOWS
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE knflows (idymax, wsnde1, wsnde2, wsnde3, wsndes,
     & wsndese, lvseep_to_c6, istr_count_knflw, tseep_divers,
     & g155_flow_from_3a, g155_pos, struc_name_meas, icnl_no_reg_seas,
```

```
     & iweir_no_reg_seas, nreg_seas, weir_reg_seas,
     & isuppflw_for_slope_kn, isdnpflw_for_slope_kn, rlvseep_to_cnl,
     $ total_small_afact,ratio_flow_s6_sta2e,ratio_flow_s6_sta2m,
     $ ratio_flow_s6_sta2w)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'STRCOUNT.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION icnl_no_reg_seas(max_n_cnl_w_weir_seas),
     & isdnpflw_for_slope_kn(max_n_str_node_dep),
     & isuppflw_for_slope_kn(max_n_str_node_dep),
     & iweir_no_reg_seas(max_n_cnl_w_weir_seas),
     & rlvseep_to_cnl(max_ncnls), tseep_divers(max_no_lvsp),
     & weir_reg_seas(max_n_cnl_w_weir_seas,max_n_weir_outf,max_n_seas),
     & total_small_afact(max_ncells)

      CHARACTER  strctrid*6, struc_name_meas(max_n_struc_meas)*6

      REAL lvseep_to_c6

      DOUBLE PRECISION dble5p5
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      CFDP = 86400. / GDAR
      IF (IDAY .NE. 1) GO TO 20
      DO 10 I = 1,istr_count_knflw
         is = istr_index(I)
         NFLO(IS,1) = 99999999
         NFLO(IS,2) = -99999999
         NFLO(IS,3) = 0
   10 CONTINUE
   20 CONTINUE
      II = 0
      NAGPTS = 0

      DO 190 I =1,ISTR_COUNT_KNFLW
         IBEG = 1
         istr_s = istr_index(i)
         II = II + 1
         IF (KFLO(ISTR_S).LT.NFLO(ISTR_S,1)) NFLO(ISTR_S,1)=KFLO(ISTR_S)
         IF (KFLO(ISTR_S).GT.NFLO(ISTR_S,2)) NFLO(ISTR_S,2)=KFLO(ISTR_S)
         NFLO(ISTR_S,3) = NFLO(ISTR_S,3) + KFLO(ISTR_S)
C
         IDV  = KFL (II,2)
```

```
         IOPT = KFL (II,1)
         Q = KFLO(ISTR_S) / IDV
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  TAKE CARE OF  S-31, S-5A, S-11  AND S-8 FLOWS
C-----------------
         call match ('S31   ',1,6,struc_name_meas,nflpts,6,imatch)
         IF (imatch .ne. 0) THEN
            strctrid = 'S31   '
         ELSE
            strctrid = name_struc_w_spec_code_wca(19)
         ENDIF
         IF (FLNM(ISTR_S) .NE. strctrid) GO TO 30
         IF (imatch .ne. 0) THEN
            qs313b = 0.0
            qs32br = 0.0
            qs32a = 0.0
            dble5p5 = dble(5.5)
            if (nint(DQU(KFL(II,6))) .eq. 0) THEN
              RMAXVOL = (chdep(KFL(II,4)) - dble5p5)*cl(KFL(II,4))
     $        *width(KFL(II,4)) + cgseep(KFL(II,4),dble5p5,0,0.)
     $        + tovflc(KFL(II,4),dble5p5,0,0.,0,1) + kflo(ifls151)*86400.
              RMAXVOL = amax1(RMAXVOL,0.0)/86400.
            else
              RMAXVOL = 99999.
            endif
            q = amin1(RMAXVOL,q)
            QU(KFL(II,4)) = QU(KFL(II,4)) - Q * 86400.
            QS313B = QS313B + Q * 86400.
            QU(KFL(II,6)) = QU(KFL(II,6)) + Q * 86400.
            upstream_inflow(KFL(II,6)) = upstream_inflow(KFL(II,6)) +
     $      Q * 86400.
            if (ivary_slope_daily_opt(KFL(II,4)) .eq. 1) then
               call match (cnm(KFL(II,4)),1,5,canal_name_dvslope,
     $         ncnl_dly_slope,5,imatch)
               dischg_for_slope_kn(imatch,2,1) =
     $         dischg_for_slope_kn(imatch,2,1) + Q
            endif
            if (ivary_slope_daily_opt(KFL(II,6)) .eq. 1) then
               call match (cnm(KFL(II,6)),1,5,canal_name_dvslope,
     $          ncnl_dly_slope,5,imatch)
               dischg_for_slope_kn(imatch,1,1) =
     $         dischg_for_slope_kn(imatch,1,1) + Q
            endif
         ELSE
            QU(KFL(II,4)) = QU(KFL(II,4)) - Q * 86400.
            if (ivary_slope_daily_opt(KFL(II,4)) .eq. 1) then
               call match (cnm(KFL(II,4)),1,5,canal_name_dvslope,
     $          ncnl_dly_slope,5,imatch)
               dischg_for_slope_kn(imatch,2,1) =
     $         dischg_for_slope_kn(imatch,2,1) + Q
            endif
            QU(KFL(II,6)) = QU(KFL(II,6)) + Q * 86400.
            if (ivary_slope_daily_opt(KFL(II,6)) .eq. 1) then
               call match (cnm(KFL(II,6)),1,5,canal_name_dvslope,
     $          ncnl_dly_slope,5,imatch)
               dischg_for_slope_kn(imatch,1,1) =
```

```
     $         dischg_for_slope_kn(imatch,1,1) + Q
            endif
            upstream_inflow(KFL(II,6)) = upstream_inflow(KFL(II,6)) + Q
     $      * 86400.
            qs32a = 0.0
            qs32br= 0.0
            qs313b= q*86400.
         ENDIF
C STORE QS313B, QS32BR AND QS32A
         JSTR=JSTR+1
         STRFLW(JSTR,IDSS) = QS313B/86400.
         JSTR=JSTR+1
         STRFLW(JSTR,IDSS) = QS32BR/86400.
         JSTR=JSTR+1
         STRFLW(JSTR,IDSS) = QS32A/86400.
C
         GO TO 190
C
C
   30    IF (FLNM(ISTR_S) .NE. 'S9    ') GO TO 40
         QSEEPG = 1.00 * rlvseep_to_cnl(KFL(II,3)) + 1. 00*
     $   rlvseep_to_cnl(KFL(II,4))
         IF (QSEEPG .GT. 0.0) THEN
            IF (Q .GT. QSEEPG) THEN
               QU(KFL(II,3))=QU(KFL(II,3)) - 1.00 * rlvseep_to_cnl
     $         (KFL(II,3)) * 86400.
               QU(KFL(II,4))=QU(KFL(II,4)) - 1.00 * rlvseep_to_cnl
     $         (KFL(II,4)) * 86400.
               QU(KFL(II,5))=QU(KFL(II,5)) - (Q - QSEEPG) * 86400.
            ELSE
               QU(KFL(II,3)) = QU(KFL(II,3)) - Q/QSEEPG * 1. 00 *
     $         rlvseep_to_cnl(KFL(II,3)) * 86400.
               QU(KFL(II,4)) = QU(KFL(II,4)) - Q/QSEEPG * 1.00 *
     $         rlvseep_to_cnl(KFL(II,4)) * 86400.
            ENDIF
         ELSE
            QU (KFL(II,5)) = QU(KFL(II,5)) - Q * 86400.
         ENDIF
         if (ivary_slope_daily_opt(KFL(II,5)) .eq. 1) then
            call match (cnm(KFL(II,5)),1,5,canal_name_dvslope,
     $       ncnl_dly_slope,5,imatch)
            dischg_for_slope_kn(imatch,2,1) =
     $      dischg_for_slope_kn(imatch,2,1) + Q
         endif
         POND(nod_ds_s9) = POND(nod_ds_s9) + Q * 86400./(GDAR*
     $   (1.0 - carfact(nod_ds_s9)))
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         GO TO 190
   40    call match ('G94AB ',1,6,struc_name_meas,nflpts,6,imatch)
         call match ('LWDD  ',1,6,struc_name_meas,nflpts,6,imatch1)
         IF (FLNM(ISTR_S) .NE. 'G94AB ' .AND. FLNM(ISTR_S) .NE. 'LWDD '
     $    .OR. (imatch .eq. 0 .and. imatch1 .eq. 0)) GO TO 50
         TOTWSNEED = WSNDE1+WSNDE2+WSNDE3+WSNDESE
         AVSUPPLY = KFLO(ISTR_S)
         IF (TOTWSNEED .GT. 0.0) THEN
            RATIOWS = AVSUPPLY / TOTWSNEED
```

```
            RATIOWS = AMIN1 (RATIOWS , 1.0)
         ELSE
            RATIOWS = 1.0
         ENDIF
         QU(ICA1) = QU(ICA1) - KFLO(ISTR_S) * 86400.
         KFLO(ISTR_S) = KFLO(ISTR_S) - AMIN1(TOTWSNEED,KFLO(ISTR_S))
C
         IF (TOTWSNEED .GT. 0.0) THEN
            QU(IE1) = QU(IE1) + (RATIOWS*WSNDE1+KFLO(ISTR_S)*0.3)*86400.
            QU(IE2) = QU(IE2) + (RATIOWS*WSNDE2+KFLO(ISTR_S)*0.5)*86400.
            QU(IE3) = QU(IE3) + (RATIOWS*WSNDE3+KFLO(ISTR_S)*0.2)*86400.
            QU(IESE) = QU(IESE) + RATIOWS * WSNDESE * 86400.
         ELSE
            QU(IE1) = QU(IE1) + KFLO(ISTR_S) * 0.3 * 86400.
            QU(IE2) = QU(IE2) + KFLO(ISTR_S) * 0.5 * 86400.
            QU(IE3) = QU(IE3) + KFLO(ISTR_S) * 0.2 * 86400.
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         GO TO 190
   50    call match ('S39   ',1,6,struc_name_meas,nflpts,6,imatch)
         IF (FLNM(ISTR_S) .NE. 'S39   ' .OR. imatch .EQ. 0)
     $   GO TO 60
         TOTWSNEED = DQU(LHL)
         AVSUPLY = KFLO(ISTR_S)
         S39EXCESS = KFLO(ISTR_S)
         IF (TOTWSNEED .GT. 0.0) THEN
            QU(LHL) = QU(LHL) + AMIN1(AVSUPLY,TOTWSNEED)*86400.
            S39EXCESS = KFLO(ISTR_S) - AMIN1(AVSUPLY,TOTWSNEED)
         ENDIF
         DQU(IS37B) = DQU(IS37B) - KFLO(IFLS38) * 0.50
         DQU(IC13) = DQU(IC13) - KFLO(IFLS38) * 0.50
         DQU(IS37B) = AMAX1(DQU(IS37B),0.0)
         DQU(IC13) = AMAX1(DQU(IC13),0.0)
         ADDWSNEED = DQU(IS37B) + DQU(IC13)
         IF (ADDWSNEED .GT. 0.0) THEN
            RATIO = S39EXCESS / ADDWSNEED
            RATIO = AMIN1(RATIO,1.0)
         ELSE
            RATIO = 0.0
         ENDIF
         QU(ICA1) = QU(ICA1) - KFLO(ISTR_S) * 86400.
         QU(IS37B) = QU(IS37B) + RATIO * DQU(IS37B) * 86400.
         QU(IC13) = QU(IC13) + RATIO * DQU(IC13) * 86400.
         QS39TBR = RATIO * (DQU(IS37B) + DQU(IC13))
         S39EXCESS = S39EXCESS - QS39TBR
         QU(LHL) = QU(LHL) + S39EXCESS * 86400.
         if (ivary_slope_daily_opt(IS37B) .eq. 1) then
            call match (cnm(IS37B),1,5,canal_name_dvslope,
     $       ncnl_dly_slope,5,imatch)
            dischg_for_slope_kn(imatch,1,1) =
     $      dischg_for_slope_kn(imatch,1,1) + RATIO * DQU(IS37B)
         endif
         if (ivary_slope_daily_opt(IC13) .eq. 1) then
            call match (cnm(IC13),1,5,canal_name_dvslope,
     $       ncnl_dly_slope,5,imatch)
            dischg_for_slope_kn(imatch,1,1) =
     $      dischg_for_slope_kn(imatch,1,1) + RATIO * DQU(IC13)
```

```
         endif
         if (ivary_slope_daily_opt(LHL) .eq. 1) then
            call match (cnm(LHL),1,5,canal_name_dvslope,
     $       ncnl_dly_slope,5,imatch)
            dischg_for_slope_kn(imatch,1,1) =
     $      dischg_for_slope_kn(imatch,1,1) + S39EXCESS
         endif
         GO TO 190
   60    IF (FLNM(ISTR_S) .NE. name_struc_w_spec_code_wca(9)) GO TO 70
c        call match('S339  ',1,6,struc_name_meas
c    $    ,nflpts,6,imatch)
c        IF (imatch .ne. 0 .or. (imatch .eq. 0 .and. runmode .eq.
c    $        'SIMUL')) THEN
c           head_drop_s11 = 0.85 * hdc(ica3,iseas)
c        ELSE
c           head_drop_s11 = 0.70 * hdc(ica3,iseas)
c        ENDIF
         QU (KFL(II,3)) = QU(KFL(II,3)) - Q * 86400.
         if (ivary_slope_daily_opt(KFL(II,3)) .eq. 1) then
            call match (cnm(KFL(II,3)),1,5,canal_name_dvslope,
     $       ncnl_dly_slope,5,imatch)
            dischg_for_slope_kn(imatch,2,2) =
     $      dischg_for_slope_kn(imatch,2,2) - 0.60*Q
            dischg_for_slope_kn(imatch,2,3) =
     $      dischg_for_slope_kn(imatch,2,3) + 0.40*Q
         endif
c        IF (chdep(ica3) + head_drop_s11 .LT. ells(nod_ds_s11ab)) THEN

            QU (KFL(II,4)) = QU(KFL(II,4)) + Q * 86400.
            if (ivary_slope_daily_opt(KFL(II,4)) .eq. 1) then
               call match (cnm(KFL(II,4)),1,5,canal_name_dvslope,
     $          ncnl_dly_slope,5,imatch)
               dischg_for_slope_kn(imatch,1,2) =
     $         dischg_for_slope_kn(imatch,1,2) + 0.80*Q
               dischg_for_slope_kn(imatch,1,3) =
     $         dischg_for_slope_kn(imatch,1,3) - 0.20*Q
            endif
            upstream_inflow(KFL(II,4)) = upstream_inflow(KFL(II,4)) + Q
     $      * 86400.
c        ELSE
c           if (ivary_slope_daily_opt(KFL(II,4)) .eq. 1) then
c              call match (cnm(KFL(II,4)),1,5,canal_name_dvslope,
c    $          ncnl_dly_slope,5,imatch)
c              if (q .lt. 1000.) then
c                 frac_f1 = 0.75
c                 frac_f2 = 0.25
c              elseif (q .lt. 2000.) then
c                 frac_f1 = 0.65
c                 frac_f2 = 0.35
c              elseif (q .lt. 3000.) then
c                 frac_f1 = 0.55
c                 frac_f2 = 0.45
c              else
c                 frac_f1 = 0.51
c                 frac_f2 = 0.49
c              endif
c              dischg_for_slope_kn(imatch,1,2) =
```

```
c    $         dischg_for_slope_kn(imatch,1,2) + frac_f1*Q
c              dischg_for_slope_kn(imatch,1,3) =
c    $         dischg_for_slope_kn(imatch,1,3) - frac_f2*Q
c           endif
c           pond(nod_ds_s11ab)=pond(nod_ds_s11ab)+0.66*q*86400./(gdar*
c    $      (1.0 - carfact(nod_ds_s11ab)))
c           pond(nod_ds_s11c) =pond(nod_ds_s11c) +0.34*q*86400./(gdar*
c    $      (1.0 - carfact(nod_ds_s11c)))
c        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         GO TO 190
   70    IF (FLNM(ISTR_S) .NE. 'G155  ' .OR. istaopt(6) .EQ.
     $    0) GO TO 72
         g155_flow_from_3a = 0.0
         K1K2 = KFL(II,3) - MINX(KFL(II,4)) +  1+ ISUM(KFL(II,4))
         IF (Q .LT. 0.0) THEN
            RCHG(K1K2) = RCHG(K1K2) + Q * CFDP /(1.0 - carfact(K1K2))
            g155_flow_from_3a = abs(q)
            g155_pos = 0.0
         ELSE
            g155_pos = q
         ENDIF

         GO TO 190
   72    IF (FLNM(ISTR_S) .NE. 'S6    ' .OR. istaopt(4) .EQ.
     $    0) GO TO 75
            NAGPTS = NAGPTS + 1
            K1 = ABS(KFL(II,4)) - MINX(ABS(KFL(II,5))) + 1 + ISUM(ABS
     $      (KFL(II,5)))
            IF (runmode .EQ. 'CALIB' .AND. KFL(II,3) .NE. 10 .AND.
     $      KFL(II,3) .NE. 40) IAGFLO(NAGPTS,1) = NINT(Q) * KFL(II,6)
     $      + IAGFLO(NAGPTS,1)
            call match (name_res_spec_code(2),1,6,resname,ntotres,6
     $       ,imatch_sta2e)
            call match (name_res_spec_code(20),1,6,resname,ntotres,6
     $       ,imatch_sta2w)
            call match (name_res_spec_code(22),1,6,resname,ntotres,6
     $       ,imatch_sta2m)
            if (imatch_sta2e .ne. 0) then
               res_area_sta2e = res_area(imatch_sta2e)
            else
               res_area_sta2e = 0.0
            endif
            if (imatch_sta2w .ne. 0) then
               res_area_sta2w = res_area(imatch_sta2w)
            else
               res_area_sta2w = 0.0
            endif
            if (imatch_sta2m .ne. 0) then
               res_area_sta2m = res_area(imatch_sta2m)
            else
               res_area_sta2m = 0.0
            endif
            sum_res_area_sta2 = res_area_sta2e+res_area_sta2w
     $      +res_area_sta2m
            if (imatch_sta2e .ne. 0) then
               ratio_flow_s6_sta2e = res_area(imatch_sta2e)
```

```
  $         /sum_res_area_sta2
         else
            ratio_flow_s6_sta2e = 0.0
         endif
         if (imatch_sta2w .ne. 0) then
            ratio_flow_s6_sta2w = res_area(imatch_sta2w)
  $         /sum_res_area_sta2
         else
            ratio_flow_s6_sta2w = 0.0
         endif
         if (imatch_sta2m .ne. 0) then
            ratio_flow_s6_sta2m = res_area(imatch_sta2m)
  $         /sum_res_area_sta2
         else
            ratio_flow_s6_sta2m = 0.0
         endif
         ista1m_loc = 25 - minx(44) + 1 + isum(44)
         if (imatch_sta2m .ne. 0) then
           ista1w_loc = 24 - minx(44) + 1 + isum(44)
         else
           ista1w_loc = 25 - minx(44) + 1 + isum(44)
         endif

         IF (Q*KFL(II,6) .LT. 0.0) THEN
            POND(K1)=POND(K1)-86400.*Q*KFL(II,6)*ratio_flow_s6_sta2e
  $         /(GDAR *(1.0 - carfact(K1) - total_small_afact(K1)))
            POND(ista1m_loc)=POND(ista1m_loc)-86400.*Q*KFL(II,6)
  $         *ratio_flow_s6_sta2m/(GDAR *(1.0 - carfact(ista1m_loc)
  $         -total_small_afact(ista1m_loc)))
            POND(ista1w_loc)=POND(ista1w_loc)-86400.*Q*KFL(II,6)
  $         *ratio_flow_s6_sta2w/(GDAR *(1.0 - carfact(ista1w_loc)
  $         -total_small_afact(ista1w_loc)))
         ELSE
            RCHG(K1)=RCHG(K1)-86400.*Q*KFL(II,6)*ratio_flow_s6_sta2e
  $         /(GDAR *(1.0 - carfact(K1) - total_small_afact(K1)))
            RCHG(ista1m_loc)=RCHG(ista1m_loc)-86400.*Q*KFL(II,6)
  $         *ratio_flow_s6_sta2m/(GDAR *(1.0 - carfact(ista1m_loc)
  $         -total_small_afact(ista1m_loc)))
            RCHG(ista1w_loc)=RCHG(ista1w_loc)-86400.*Q*KFL(II,6)
  $         *ratio_flow_s6_sta2w/(GDAR *(1.0 - carfact(ista1w_loc)
  $         -total_small_afact(ista1w_loc)))
         ENDIF
      GO TO 190
75    IF (FLNM(ISTR_S) .NE. 'S332B ') GO TO 80
      qu(KFL(II,3)) = qu(KFL(II,3)) - q * 86400.
      if (ivary_slope_daily_opt(KFL(II,3)) .eq. 1) then
         call match (cnm(KFL(II,3)),1,5,canal_name_dvslope,
  $       ncnl_dly_slope,5,imatch)
         dischg_for_slope_kn(imatch,2,2) =
  $      dischg_for_slope_kn(imatch,2,2) - Q
      endif
      call match ('S332BR',1,6,resname,ntotres,6,imatch_s332br)
      IF (imatch_s332br .ne. 0) THEN
        K2K3 = KFL(II,4) - MINX(KFL(II,5)) + 1 + ISUM(KFL(II,5))
        IF (stage_res(K2K3,1) .GT. ells_in_res(K2K3,1)) THEN
          pond_in_res(K2K3,1) = pond_in_res(K2K3,1) + q*86400.
  $        /(sfactor(imatch_s332br) * gdar)
```

```
              stage_res(K2K3,1) = ells_in_res(K2K3,1)+pond_in_res(K2K3,1)
            ELSE
              dpth_below_ls_res = (ells_in_res(K2K3,1)-stage_res(K2K3,1))
     $        *s(K2K3)
              dpth_of_inflw = q * 86400. / (sfactor(imatch_s332br) *gdar)
              stage_res(K2K3,1) = stage_res(K2K3,1) + amin1(dpth_of_inflw
     $        ,dpth_below_ls_res)/s(K2K3) + amax1(dpth_of_inflw
     $        -dpth_below_ls_res,0.0)
              pond_in_res(K2K3,1) = amax1(stage_res(K2K3,1) - ells_in_res
     $        (K2K3,1),0.0)
            ENDIF
            res_struc_inflw(imatch_s332br) = res_struc_inflw
     $      (imatch_s332br) + q * 1.9835
          ELSE
             pond(K2K3) = pond(K2K3) + q * 86400. / (gdar * (1.0
     $       - carfact(k2k3) - total_small_afact(k2k3)))
          ENDIF
          GO TO 190
C-----------------
   80     DO 180 J = IBEG,IDV
             IF (J .EQ. 1) GO TO 90
             II = II + 1
             IOPT = KFL (II,1)
             IDV  = KFL (II,2)
   90        GO TO (100,110,120,130,140,150,160,170) IOPT
             PRINT *,'INVALID KNOWN FLOW OPTION ',FLNM(ISTR_S),' ',IOPT ,
     $       IDV
             STOP 77
  100        K1K2 = KFL(II,3) - MINX(KFL(II,4)) +  1+ ISUM(KFL(II,4))
             rem_area_in_cell = 1.0 / (1.0 - carfact(k1k2)
     $        - total_small_afact(k1k2))
             IF (NINT(Q) .GT. 0) THEN
                POND(K1K2) = POND(K1K2) + Q * CFDP * rem_area_in_cell
             ELSE
                SGN = -1.0
                outf_greater_than_pond = amax1(sgn* Q * CFDP
     $          * rem_area_in_cell - POND(K1K2) ,0.0)
                POND(K1K2) = POND(K1K2) - AMIN1(sgn* Q * CFDP
     $          * rem_area_in_cell,POND(K1K2))
                RCHG(K1K2) = RCHG(K1K2) - outf_greater_than_pond
             ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             GO TO 180
  110        QU(KFL(II,3)) = QU(KFL(II,3)) + Q * 86400.
             IF (ivary_slope_daily_opt(KFL(II,3)) .eq. 1) THEN

                call match(cnm(KFL(II,3)),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
                sgn = 1
                IF (isdnpflw_for_slope_kn(i) .eq. 3) then
                   sgn = -1
                ENDIF
                dischg_for_slope_kn(imatch_slope,1,isdnpflw_for_slope_kn
     &          (i)) = dischg_for_slope(imatch_slope,1,
     &          isdnpflw_for_slope_kn(i)) + sgn*q
              ENDIF
             GO TO 180
```

```
120       K1K2 = KFL(II,3) - MINX(KFL(II,4)) + 1 + ISUM(KFL(II,4))
          K3K4 = KFL(II,5) - MINX(KFL(II,6)) + 1 + ISUM(KFL(II,6))
          POND(K1K2) = POND(K1K2) - Q * CFDP / (1.0 - carfact(K1K2)
   &      - total_small_afact(K1K2))
          POND(K3K4) = POND(K3K4) + Q * CFDP / (1.0 - carfact(K3K4)
   &      - total_small_afact(K3K4))
          GO TO 180
130       K1K2 = KFL(II,3) - MINX(KFL(II,4)) + 1 + ISUM(KFL(II,4))
          POND(K1K2) = POND(K1K2) - Q * CFDP / (1.0 - carfact(K1K2)
   $       - total_small_afact(K1K2))
          QU(KFL(II,5)) = QU(KFL(II,5))  + Q * 86400.
          upstream_inflow(KFL(II,5)) = upstream_inflow(KFL(II,5)) + Q
   $      * 86400.
          IF (ivary_slope_daily_opt(KFL(II,5)) .eq. 1) THEN

             call match(cnm(KFL(II,5)),1,5,canal_name_dvslope
   &          ,ncnl_dly_slope,5,imatch_slope)
             sgn = 1
             IF (isdnpflw_for_slope_kn(i) .eq. 3) then
                sgn = -1
             ENDIF
             dischg_for_slope_kn(imatch_slope,1,isdnpflw_for_slope_kn
   &         (i)) = dischg_for_slope_kn(imatch_slope,1,
   &         isdnpflw_for_slope_kn(i)) + sgn*q
          ENDIF
          GO TO 180
140       K2K3 = KFL(II,4) - MINX(KFL(II,5)) + 1 + ISUM(KFL(II,5))
          QU(KFL(II,3)) = QU(KFL(II,3)) - Q * 86400.
          POND(K2K3) = POND(K2K3) + Q * CFDP / (1.0 - carfact(K2K3)
   &       - total_small_afact(K2K3))
          IF (ivary_slope_daily_opt(KFL(II,3)) .eq. 1) THEN
            call match(cnm(KFL(II,3)),1,5,canal_name_dvslope
   &         ,ncnl_dly_slope,5,imatch_slope)
            sgn = 1
            IF (isuppflw_for_slope_kn(i) .eq. 2) then
               sgn = -1
            ENDIF
            IF (FLNM(istr_s) .eq. 'S144  ' .or. FLNM(istr_s) .eq.
   &        'S145  ' .or. FLNM(istr_s) .eq. 'S146  ') THEN
               frac1_1 = 0.90
               frac2_1 = 0.10
            ELSE
               frac1_1 = 1.0
               frac2_1 = 0.0
            ENDIF
            dischg_for_slope_kn(imatch_slope,2,isuppflw_for_slope_kn
   &        (i)) = dischg_for_slope_kn(imatch_slope,2,
   &        isuppflw_for_slope_kn(i)) + sgn*frac1_1*q
            dischg_for_slope_kn(imatch_slope,2,2) =
   &        dischg_for_slope_kn(imatch_slope,2,2) - frac2_1*q
          ENDIF
          GO TO 180
150       QU(KFL(II,3)) = QU(KFL(II,3)) - Q * 86400.
          QU(KFL(II,4)) = QU(KFL(II,4)) + Q * 86400.
          IF (ivary_slope_daily_opt(KFL(II,3)) .eq. 1) THEN

             call match(cnm(KFL(II,3)),1,5,canal_name_dvslope
```

```
&          ,ncnl_dly_slope,5,imatch_slope)
           sgn = 1
           IF (isuppflw_for_slope_kn(i) .eq. 2) then
              sgn = -1
           ENDIF
           IF (FLNM(istr_s) .eq. 'S12A  ' .or. FLNM(istr_s) .eq.
&          'S12B ' .or. FLNM(istr_s) .eq. 'S12C  ' .or. FLNM(istr_s)
&          .eq. 'S12D  ' .OR. FLNM(istr_s) .eq. 'S333  ') THEN
            totflow = kflo(ifls333)+kflo(ifls12a)+kflo(ifls12b)
&           +kflo(ifls12c)+kflo(ifls12d)
            if (totflow .lt. 500.) then
               frac_flw_slpe1 = 1.0
               frac_flw_slpe2 = 0.0
               frac_flw_slpe3 = 0.0
            elseif (totflow .lt. 1500.) then
               frac_flw_slpe1 = 0.25
               frac_flw_slpe2 = 0.65
               frac_flw_slpe3 = 0.1
            elseif (totflow .lt. 2500.) then
               frac_flw_slpe1 = 0.00
               frac_flw_slpe2 = 0.58
               frac_flw_slpe3 = 0.42
            elseif (totflow .lt. 4000.) then
               frac_flw_slpe1 = 0.0
               frac_flw_slpe2 = 0.52
               frac_flw_slpe3 = 0.48
            else
               frac_flw_slpe1 = 0.0
               frac_flw_slpe2 = 0.51
               frac_flw_slpe3 = 0.49
            endif
            dischg_for_slope_kn(imatch_slope,2,isuppflw_for_slope_kn
&            (i)) = dischg_for_slope_kn(imatch_slope,2,
&            isuppflw_for_slope_kn(i)) + sgn*frac_flw_slpe1*q
            dischg_for_slope_kn(imatch_slope,2,3) =
&           dischg_for_slope_kn(imatch_slope,2,3) + frac_flw_slpe2*q
            dischg_for_slope_kn(imatch_slope,2,2) =
&           dischg_for_slope_kn(imatch_slope,2,2) - frac_flw_slpe3*q
           ELSE
            dischg_for_slope_kn(imatch_slope,2,isuppflw_for_slope_kn
&           (i)) = dischg_for_slope_kn(imatch_slope,2,
&           isuppflw_for_slope_kn(i)) + sgn*q
           ENDIF
        ENDIF
        IF (ivary_slope_daily_opt(KFL(II,4)) .eq. 1) THEN

           call match(cnm(KFL(II,4)),1,5,canal_name_dvslope
&           ,ncnl_dly_slope,5,imatch_slope)
           sgn = 1
           IF (isdnpflw_for_slope_kn(i) .eq. 3) then
              sgn = -1
           ENDIF
           dischg_for_slope_kn(imatch_slope,1,isdnpflw_for_slope_kn
&          (i)) = dischg_for_slope_kn(imatch_slope,1,
&          isdnpflw_for_slope_kn(i)) + sgn*q
        ENDIF
        upstream_inflow(KFL(II,4)) = upstream_inflow(KFL(II,4)) + Q
```

```
     $      * 86400.
            GOTO 180
  160       NAGPTS = NAGPTS + 1
            IF (runmode .EQ. 'CALIB'.AND. KFL(II,3) .NE. 10 .AND. KFL
     $      (II,3) .NE. 40) IAGFLO(NAGPTS,1) = NINT(Q) * KFL(II,5) +
     $      IAGFLO(NAGPTS,1)

            QU(ABS(KFL(II,4)))=QU(ABS(KFL(II,4))) - 86400. * Q * KFL
     $      (II,5)
            IF (ivary_slope_daily_opt(ABS(KFL(II,4))) .eq. 1) THEN

               call match(cnm(ABS(KFL(II,4))),1,5,canal_name_dvslope
     &          ,ncnl_dly_slope,5,imatch_slope)
               sgn = 1
               IF (isdnpflw_for_slope_kn(i) .eq. 3) then
                  sgn = -1
               ENDIF
               dischg_for_slope_kn(imatch_slope,1,isdnpflw_for_slope_kn
     &         (i)) = dischg_for_slope_kn(imatch_slope,1,
     &         isdnpflw_for_slope_kn(i)) + sgn*q
            ENDIF
            GO TO 180
  170       NAGPTS = NAGPTS + 1
            K1 = ABS(KFL(II,4)) - MINX(ABS(KFL(II,5))) + 1 + ISUM(ABS
     $      (KFL(II,5)))
            IF (runmode .EQ. 'CALIB' .AND. KFL(II,3) .NE. 10 .AND.
     $      KFL(II,3) .NE. 40) IAGFLO(NAGPTS,1) = NINT(Q) * KFL(II,6)
     $      + IAGFLO(NAGPTS,1)

            IF (Q*KFL(II,6) .LT. 0.0) THEN
               POND(K1) = POND(K1) - 86400. * Q * KFL(II,6) / (GDAR
     $         *(1.0 - carfact(K1) - total_small_afact(K1)))

            ELSE
               RCHG(K1) = RCHG(K1) - 86400. * Q * KFL(II,6) / (GDAR
     $         *(1.0 - carfact(K1) - total_small_afact(K1)))

            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
  180    CONTINUE
C
  190 CONTINUE
c
      IF (ISTAOPT(10) .EQ. 1) THEN
         NAGPTS = NAGPTS + 1
      ENDIF
C<<
C-----------------
C  SET TODAYS REGULATION SCHEDULE WHERE NECESSARY
C-----------------
      IF ( NREG_SEAS .GT. 0 ) THEN
         DO 210 I = 1,NREG_SEAS
             CREL(icnl_no_reg_seas(I),iweir_no_reg_seas(I))
     $       = weir_reg_seas(I,iweir_no_reg_seas(I),iseas)
  210    CONTINUE
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

```
C
      RETURN
      END
```

lake_nonreg_wca.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lake_nonreg_wca.F,v 1.25 2005/06/10 13:29:09 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lake_nonreg_wca.F,v $
C   $Name: rel-5-5 $
C
C
C  SUBROUTINE TO CALCULATE THE WATER SUPPLY RELEASES FROM LAKE
C  OKEECHOBEE TO THE WCAs TO MEET ENVIRONMENTAL (PRIMARILY
C  EVERGLADES) AND URBAN NEEDS DURING APPROPRIATE TIMES
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lake_nonreg_wca (dellec, crmin, volwtr, numd, wsnde1,
     & wsnde2, wsnde3, wsndes, wsndese, wsneedl8, dmds4, dmds236,
     & s65ernff, dmdsemind, jday, pctmet, ws_from_lo_s6,
     & s236so, rlok_to_sa1, outflow_from_sta2_for_ws,
     & runoff_298_districts, wsuply, tcnsrnff, divers,
     & out_to_wca1_frm_sta1w, out_to_wca1_frm_sta1e, runoff_south,
     & ifacteaa, stgref_asr, stgref, ws_from_lo_s5a, isem_flg,
     & sem_dmnd_orig, dmnd_sugh, ws_frm_lok_to_semcyp,
     & ws_frm_lok_to_sugh, dmnd_sugh_orig, capac_semws, capac_sugh,
     & stgref_asr_rec, runoff_s7, runoff_s150, avg_lokstage,
     & stagelo_beg, dmnd_nnr_aft_resasr, res_outflow_to_basin,
     & avg_sim_wca_stg_floor, sem_dmnd_met_bef_lok, ires_no,
     & water_supply_to_holyl, lok_criteria_wca1_lok_ws, climate_index,
     & norstorrf, norstorpet, frac_env_met, tcrnbsres_to_lok,
     & frac_thru_s8_sta34, ws_to_roten_frm_lok, fplresws,
     & ws_to_sta34_frm_lok, rloc_excess_for_ws, max_days_mean, sumrain,
     & tot_flw_to_c9r_frm_nlkblt, ro_factor, sprev_res, s1,
     & dmnd_lec_s5a, dmnd_lec_s6, dmnd_lec_sa2, stg_backflw_stl,
     & stgref_pulse_bot, stgref_nstor_to_lok, stgref_lok_to_nstor,
     & vol_flow_res, runoff_south_bef_divers, stgref_lok_to_eaasurgt,
     & stgref_regtoest, kfloref, nmdays_lok, ws_from_acmres_lwdd,
     & issm_cutb_opt_bcyp, pot_del_to_eaa_basin, alocnnr_potdel,
```

```
& total_lec_dmnd_lok, issm_cutb_opt_brghton,
& g94abcapac, rlvseep_frm_cnl, ws_bkp_cap, iwsbpcntr, maxbpcntr,
& iwsbkpwsta , bflo_cap_l8, iroute_to_wca3a_frm_holy,
& igravity_s8_nosta, total_small_afact,runoff_to_sta,s4bsn_to_lok,
& uisrnff,iday_lok_in_ssm,bmp_ssm_opt,stgref_state_eaares,
& iopt_env_eaar_thru_g372)

 INCLUDE 'wmm_par.inc'
 INCLUDE 'abc.inc'
 INCLUDE 'agdata.inc'
 INCLUDE 'c1.inc'
 INCLUDE 'c2.inc'
 INCLUDE 'routc.inc'
 INCLUDE 'stat.inc'
 INCLUDE 'wrout.inc'
 INCLUDE 'stas.inc'
 INCLUDE 'wcaot.inc'
 INCLUDE 'lakwca.inc'
 INCLUDE 'cstlucie.inc'
 INCLUDE 'ccaloos.inc'
 INCLUDE 'environ.inc'
 INCLUDE 'asr_param.inc'
 INCLUDE 'asr1.inc'
 INCLUDE 'eaa_caps.inc'
 INCLUDE 'resws.inc'
 INCLUDE 'opersched.inc'
 INCLUDE 'minfllvl.inc'
 INCLUDE 'flw298dso.inc'
 INCLUDE 'loktarg.inc'
 INCLUDE 'eaa_pump_caps.inc'
 INCLUDE 'wcasim.inc'
 INCLUDE 'lokres.inc'
 INCLUDE 'climvar.inc'
 INCLUDE 'resbudg.inc'

 DIMENSION alocnnr_potdel(2),
& avg_sim_wca_stg_floor(max_n_wcas,max_n_minfllvl_loc),
& climate_index(12), crmin(max_ncnls,max_n_seas), dmdsemind(366),
& fplresws(366), ires_no(max_n_res), lwsbkp_floor(max_n_eaa_condt),
& numd(12), pot_del_to_eaa_basin(max_n_eaa_condt,2),
& rem_weir_capac(max_n_eaa_condt),total_small_afact(max_ncells),
& res_outflow_to_basin(max_n_res,max_nostr_res),
& rloc_excess_for_ws(max_n_serv_area), rlokstage(10),
& rlvseep_frm_cnl(max_ncnls), rmax_capac_ws_bkp(max_n_eaa_condt),
& ro_factor(12), runoff_south(max_n_eaa_condt),
& runoff_south_bef_divers(max_n_eaa_condt), s65ernff(366),
& sumrain(max_ncells), tcnsrnff(366), volwtr(max_n_res),
& ws_bkp_cap(max_n_eaa_condt,2), wsuply(max_n_res),uisrnff(366),
& LWSLECSTA(max_n_eaa_condt)

 CHARACTER cname*6, cname_lower*6, cname_upper*6, str_name*6,
& str_name_new*6, str_name_new_pmp*6,bmp_ssm_opt*3

 LOGICAL lwsbkp_floor, lwsbpmp, lwswca, LWSLECSTA

 REAL darefet, infltores, kbfbps2, kfloref(max_n_eaa_condt),
& miamic_cap, miamic_cap_lower, miamic_cap_wsdeltosta, nnrc_cap,
```

```
     & nnrc_cap_lower, nnrc_cap_wstosta, norstorpet(366), norstorrf(366)

      DOUBLE PRECISION crmin,floor,rmin_stg_wca1
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (day1) THEN
         lwsbpmp = .FALSE.
         avg_lokstage = stagelo
      ENDIF

      IF (lok_split_option .eq. 0) THEN
         SPREV = STG2STO(STAGELO,0)
      ELSE
         SPREV = STG2STOLITZ(STAGELO,0)
         sprev_res = stg2stores(stagelo_res,0)
      ENDIF

      call bpts(iday,month,10.0,1,n_fc_zones,1,stgref)
      call bpts(iday,month,10.0,1,n_fc_zones+4,1,stgref_asr_rec)
      call bpts(iday,month,10.0,1,n_fc_zones+1,1,stgref_asr)
      call bpts(iday,month,10.0,1,n_fc_zones+2,1,stgref_lok_to_nstor)
      call bpts(iday,month,10.0,1,n_fc_zones+5,1,stgref_nstor_to_lok)
      call bpts(iday,month,10.0,1,n_fc_zones+3,1,stgref_lok_to_eaasurgt)
      call bpts(iday,month,10.0,1,izone_bot_flood_flows_south,1
     $              ,stgref_regtoest)
      call bpts(iday,month,10.0,1,izone_bot_state_eaares,1
     $              ,stgref_state_eaares)
      call bpts(iday,month,10.0,1,n_fc_zones+8,1,stg_backflw_stl)
      call bpts(iday,month,10.0,1,n_fc_zones+10,1,upper_stg_ws_bkp)
      call bpts(iday,month,10.0,1,n_fc_zones+11,1,rlower_stg_ws_bkp)
      if (no_puls_zones .gt. 0) then
        call bpts(iday,month,10.0,1,mzone_puls_bot,1
     $              ,stgref_pulse_bot)
      endif
c
c  determine current water supply backpumping zone for LOK
c
      if (stagelo .le. rlower_stg_ws_bkp) then
         do ieaac = 1,neaabsn-1
            rmax_capac_ws_bkp(ieaac) = ws_bkp_cap(ieaac,2)
         enddo
      elseif (stagelo .le. upper_stg_ws_bkp) then
         do ieaac = 1,neaabsn-1
            rmax_capac_ws_bkp(ieaac) = ws_bkp_cap(ieaac,1)
         enddo
      else
         do  ieaac = 1,neaabsn-1
            rmax_capac_ws_bkp(ieaac) = 0.0
         enddo
      endif
c
c time dependent floor elevation for WCAs
c
      do iwca = 1,nwcas
         if (ifloor_line_indx(iwca+1) .gt. 0) then
           do i = 1,n_intcnl(iwca)
             rmnstg(iwca,i) = regstg_wca(iwca,ifloor_line_indx(iwca+1))
```

```
            enddo
          endif
       enddo
c
c  set condition for change in ws backpumping to LOK from normal
c  operations
c
       LWSBKP_floor(is8bsn-6) = chdep(intcnl(iwca3a,
     $ iconvey_canal_no_wca(iwca3a))) .LT.
     $ rmnstg(idn_wca_indx(is8bsn-6),iconvey_canal_no_wca(iwca3a))
     $ + 0.05
       LWSBKP_floor(is7bsn-6) = trig_wca_stg(idn_wca_indx(is7bsn-6))
     $ .LT. rmnstg(idn_wca_indx(is7bsn-6),iconvey_canal_no_wca
     $ (iwca2a)) + 0.05
       LWSBKP_floor(is6bsn-6) = trig_wca_stg(idn_wca_indx(is6bsn-6))
     $ .LT. rmnstg(idn_wca_indx(is6bsn-6),iconvey_canal_no_wca
     $ (iwca1)) + 0.05
       LWSBKP_floor(is5absn-6) = trig_wca_stg(idn_wca_indx(is5absn-6))
     $ .LT. rmnstg(idn_wca_indx(is5absn-6),iconvey_canal_no_wca
     $ (iwca1)) + 0.05
c
c   set condition for STA treatment of water supply deliveries to LEC
c    service areas
c
       LWSLECSTA(is8bsn-6) = chdep(intcnl(iwca3a,
     $ iconvey_canal_no_wca(iwca3a))) .GT.
     $ rmnstg(idn_wca_indx(is8bsn-6),iconvey_canal_no_wca(iwca3a))
     $ + 0.05
       LWSLECSTA(is7bsn-6) = trig_wca_stg(idn_wca_indx(is7bsn-6))
     $ .GT. rmnstg(idn_wca_indx(is7bsn-6),iconvey_canal_no_wca
     $ (iwca2a)) + 0.05
       LWSLECSTA(is6bsn-6) = trig_wca_stg(idn_wca_indx(is6bsn-6))
     $ .GT. rmnstg(idn_wca_indx(is6bsn-6),iconvey_canal_no_wca
     $ (iwca1)) + 0.05
       LWSLECSTA(is5absn-6) = trig_wca_stg(idn_wca_indx(is5absn-6))
     $ .GT. rmnstg(idn_wca_indx(is5absn-6),iconvey_canal_no_wca
     $ (iwca1)) + 0.05
c
c  set up fraction of expected delivery to be made from WCAs
c
        iday_of_week = idofw(month,iday,ifyr+iyear-1)
        frac_del_frm_wca(iwca3a) = 0.0
        frac_del_frm_wca(iwca2a) = 0.0
        frac_del_frm_wca(iwca1) = 0.0
c  WCA-3A
        IF ((chdep(intcnl(iwca3a,iconvey_canal_no_wca(iwca3a)))
     $  .LT. rmnstg(idn_wca_indx(is8bsn-6),iconvey_canal_no_wca
     $  (iwca3a)) + 0.05) .OR. LFRACBFLR(iwca3a))
     $  THEN
            LFRACBFLR(iwca3a) = .TRUE.
            DO iw = 1, no_of_days_wk_wcadel(iwca3a,2)
               IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca3a
     $         ,iw,2)) THEN
                  frac_del_frm_wca(iwca3a) = frac_wcaws_del(iwca3a,2)
               ENDIF
            ENDDO
        ELSE
```

```
            DO iw = 1, no_of_days_wk_wcadel(iwca3a,1)
              IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca3a
     $        ,iw,1)) THEN
                 frac_del_frm_wca(iwca3a) = frac_wcaws_del(iwca3a,1)
              ENDIF
            ENDDO
        ENDIF

        LFRACBFLR(iwca3a) = .NOT. (chdep(intcnl(iwca3a,
     $  iconvey_canal_no_wca(iwca3a))) .GE. rmnstg
     $  (idn_wca_indx(is8bsn-6),iconvey_canal_no_wca
     $  (iwca3a)) + 0.25)
c  WCA-2A
        IF ((trig_wca_stg(iwca2a) .LT. rmnstg(iwca2a
     $  ,iconvey_canal_no_wca(iwca2a))+0.05) .OR. LFRACBFLR(iwca2a))
     $  THEN
           LFRACBFLR(iwca2a) = .TRUE.
           DO iw = 1, no_of_days_wk_wcadel(iwca2a,2)
              IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca2a
     $        ,iw,2)) THEN
                 frac_del_frm_wca(iwca2a) = frac_wcaws_del(iwca2a,2)
              ENDIF
           ENDDO
        ELSE
           DO iw = 1, no_of_days_wk_wcadel(iwca2a,1)
              IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca2a
     $        ,iw,1)) THEN
                 frac_del_frm_wca(iwca2a) = frac_wcaws_del(iwca2a,1)
              ENDIF
           ENDDO
        ENDIF
        LFRACBFLR(iwca2a) = .NOT. (trig_wca_stg(iwca2a) .GE.
     $   rmnstg(iwca2a,iconvey_canal_no_wca(iwca2a)) + 0.25)
c  WCA-1
        IF ((trig_wca_stg(iwca1) .LT. rmnstg(iwca1
     $   ,iconvey_canal_no_wca(iwca1))+0.05) .OR. LFRACBFLR(iwca1)) THEN
           LFRACBFLR(iwca1) = .TRUE.
           DO iw = 1, no_of_days_wk_wcadel(iwca1,2)
              IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca1
     $        ,iw,2)) THEN
                 frac_del_frm_wca(iwca1) = frac_wcaws_del(iwca1,2)
              ENDIF
           ENDDO
        ELSE
           DO iw = 1, no_of_days_wk_wcadel(iwca1,1)
              IF (iday_of_week .EQ. idays_of_wk_wcadel_indx(iwca1
     $        ,iw,1)) THEN
                 frac_del_frm_wca(iwca1) = frac_wcaws_del(iwca1,1)
              ENDIF
           ENDDO
        ENDIF
        LFRACBFLR(iwca1) = .NOT. (trig_wca_stg(iwca1) .GE.
     $   rmnstg(iwca1,iconvey_canal_no_wca(iwca1))+0.25)
c
c
       LWSWCA = .FALSE.
       ws_to_lec_s354 = 0.0
```

```
      ws_to_lec_s351 = 0.0
      ws_to_lec_s352 = 0.0
      ws_to_lec_s351_nnrc  = 0.0
      ws_to_lec_s351_hill  = 0.0
      ws_to_lec_s351_wca3a = 0.0
      env_ws_from_res_thru_g372 = 0.0
      total_env_ws_from_eaares = 0.0
      emerg_bypass_s7 = 0.0
      emerg_bypass_s8 = 0.0
      dmnd_lec_sa3_s150 = 0.0
      DELLEC = 0.0
      WSS351 = 0.0
      WSS352 = 0.0
      REGS351 = 0.0
      REGS352 = 0.0
      REGS354 = 0.0
      twsuply = 0.0
      nnrreg  = 0.0
      hlsbreg = 0.0
      s150_wsdel = 0.0
      ws_from_lo_s5a = 0.0
      flow_to_import = 0.0
      flow_to_import_miami = 0.0
      flow_to_import_nnrc  = 0.0
      flow_to_import_hill  = 0.0
      flow_to_import_wpb   = 0.0
      flow_s8new_to_sta    = 0.0
      flow_s7new_to_sta    = 0.0
      ws_down_s6           = 0.0
      ws_to_sta34_frm_lok  = 0.0
      ws_to_sta34_via_nnrc = 0.0
      rnorth_stor_to_lok = 0.0
      rlok_to_north_stor = 0.0
      res_outflow_env  = 0.0
      add_ws_pipeline_3a = 0.0
      total_divers_to_eaares = 0.0
      ws_pipeline_2a  = 0.0
      ws_pipeline_3a  = 0.0
      ws_pipeline_ca1 = 0.0
      ws_pipeline_ca1_hill = 0.0
      ws_pipeline_ca1_wpb = 0.0
      ws_pipeline_3a_s150 = 0.0
      rlocal_storage_3a = 0.0
      wsndes_rev  = wsndes
      wsndese_rev = wsndese
      do ib = 1,neaabsn
       flow_to_eaa_res_rnff(ib) = 0.0
       do k = 1,no_of_res_reg_frm_lok(ib)
         flow_to_eaa_res_reg(ib,k)  = 0.0
       enddo
      enddo
      inflow_indx = 1
c
      if (DRYSEAS) then
         iseas = 1
      else
         iseas = 2
```

```
      endif
c
c      chgrnfistapoga(jday) = chgrnfistapoga(jday) * ro_factor(month)
C
      DPTHGW = (ELLS(IG3A4) - H(IG3A4)) * S(IG3A4)
      IF (POND(IG3A4) .GT. DPTHGW) THEN
       STG3A4 = POND(IG3A4) - DPTHGW + ELLS(IG3A4)
      ELSE
       STG3A4 = (POND(IG3A4)-DPTHGW)/S(IG3A4) + ELLS(IG3A4)
      ENDIF
C
      DPTHGW = (ELLS(IG3A28) - H(IG3A28)) * S(IG3A28)
      IF (POND(IG3A28) .GT. DPTHGW) THEN
       STG3A28 = POND(IG3A28) - DPTHGW + ELLS(IG3A28)
      ELSE
       STG3A28 = (POND(IG3A28)-DPTHGW)/S(IG3A28) + ELLS(IG3A28)
      ENDIF
c
c  calculate running mean of simulated lake stages to determine
c  if lake is rising of falling . This is used as a condition
c  for injection of LOK water into proposed ASR
c
      if (runmode .eq. 'SIMUL' .and. max_days_mean .gt. 0) then
       nmdays_lok = nmdays_lok + 1
       if (nmdays_lok .gt. max_days_mean)
     $     nmdays_lok = max_days_mean
       if (ndays_sim .ge. max_days_mean) then
         do id = 1,nmdays_lok-1
           if (ndays_sim .gt. max_days_mean) then
             rlokstage(id) = rlokstage(id+1)
           endif
c
           avg_lokstage = avg_lokstage
     $         + rlokstage(id) / nmdays_lok
         enddo
         rlokstage(nmdays_lok) = stagelo
         avg_lokstage = avg_lokstage + stagelo / nmdays_lok
       else
         rlokstage(nmdays_lok) = stagelo
         avg_lokstage = rlokstage(nmdays_lok)
       endif
      endif
c
       do isa = 1,nwcas
        LWSLO_floor(ISA) = .FALSE.
       enddo
c
      if (floor_grid_cell_opt .eq. 'MARSH') then
       do isa = 1,nwcas
        do i = 1,no_of_subareas_floor(isa)
        if (nmpts_wca_floor(isa,i) .gt. 0) then
          avg_sim_wca_stg_floor(isa,i) = 0.0
          do j = 1,nmpts_wca_floor(isa,i)
             depthgw = (ells(icell_no_floor(isa,i,j)) -
     $        h(icell_no_floor(isa,i,j))) * s(icell_no_floor(isa,i,j))
             if (pond(icell_no_floor(isa,i,j)) .gt. depthgw) then
                wca_nodal_stage = pond(icell_no_floor(isa,i,j)) -
```

```
     $             depthgw + ells(icell_no_floor(isa,i,j))
             else
                wca_nodal_stage = (pond(icell_no_floor(isa,i,j))
     $             - depthgw)/s(icell_no_floor(isa,i,j))
     $             + ells(icell_no_floor(isa,i,j))
             endif
             avg_sim_wca_stg_floor(isa,i) = avg_sim_wca_stg_floor(isa,i)
     $             + wca_nodal_stage / nmpts_wca_floor(isa,i)
          enddo
        endif
        enddo
       enddo
      endif
C
C IF STAGE IN LAKE OKEECHOBEE IS GREATER THAN 1 FOOT ABOVE WCA-1
C STAGE FOR PROPOSED SCHEDULE (FUTURE BASE AND ALTERNATIVES) ,THEN
C WATER SUPPLY WILL BE BROUGHT IN FROM LAKE OKEECHOBEE
C
       if (opt_wca1_prop.and.iconvey_canal_no(is5absn-6).ne.0) then
           if (wcaenv(idn_wca_indx_env(is5absn-6)) .eq. 'FALSE')  then
             if (stagelo .gt. trig_wca_stg(idn_wca_indx(is5absn-6))-1.0
     $        .and. trig_wca_stg(idn_wca_indx(is5absn-6)) .lt.
     $        regstg_wca(idn_wca_indx(is5absn-6),1)) then
               rmin_stg_wca1 = max(trig_wca_stg
     $         (idn_wca_indx(is5absn-6)),rmnstg(idn_wca_indx(is5absn-6)
     $               ,iconvey_canal_no(is5absn-6)))
             lok_criteria_wca1 = 1
             else
                rmin_stg_wca1 = rmnstg(idn_wca_indx(is5absn-6)
     $               ,iconvey_canal_no(is5absn-6))
             lok_criteria_wca1 = 0
             endif
           else
             rmin_stg_wca1 = rmnstg(idn_wca_indx(is5absn-6)
     $               ,iconvey_canal_no(is5absn-6))
             lok_criteria_wca1 = 0
           endif
       elseif(iconvey_canal_no(is5absn-6).ne.0) then
         rmin_stg_wca1 = rmnstg(idn_wca_indx(is5absn-6)
     $               ,iconvey_canal_no(is5absn-6))
         lok_criteria_wca1 = 0
       else
         lok_criteria_wca1 = 0
       endif
c
       rmin_stg_wca2a = rmnstg(idn_wca_indx(is7bsn-6)
     $        ,iconvey_canal_no(is7bsn-6))
       rmin_stg_wca3a = rmnstg(idn_wca_indx(is8bsn-6)
     $        ,iconvey_canal_no(is150bws-6))
c
      do i = 1,nwcas
        do j = 1,n_intcnl(i)
          if (wca_name(i) .eq. 'WCA1 ') then
            rmin_stg_wca(i,j) = rmin_stg_wca1
          else
            rmin_stg_wca(i,j) = rmnstg(i,j)
          endif
```

```
          floor_elev(i,j) = rmin_stg_wca(i,j)
        enddo
      enddo
C
C  DETERMINE L.O. STAGE FOR WATER SUPPLY BACKPUMPING
C   UPPER LIMIT FOR ZONE C OF WATER SHORTAGE OPERATIONS
C
      ZONECSTG = rlower_stg_ws_bkp
C
C COMPUTE CAPACITY CURVES FOR DISCHARGES FROM LAKE OKEECHOBEE TO
C WATER CONSERVATION AREAS
C
C TO WCA-3A
C
      if (istaopt(is8bsn-6) .eq. 1) then
        if (eaa_conv_opt_s8 .eq. 'PUMP') then
          grf3amx = eaa_convey(2,1)
        else
          grf3amx = eaa_convey(1,1)
        endif
      endif
C
C TO WCA-2A
C
       if (istaopt(is7bsn-6) .eq. 1) then
         if (eaa_conv_opt_s7 .eq. 'PUMP') then
           grf2amx = eaa_convey(5,1)
         else
           grf2amx = eaa_convey(3,1)
         endif
       endif
C
C TO WCA-1 BY HILLSBORO CANAL THROUGH S-6
C
       if (istaopt(is6bsn-6) .eq. 1) then
          if (istaopt(is7bsn-6) .eq. 1) then
             if (eaa_conv_opt_s7 .eq. 'PUMP') then
               fllks6 = eaa_convey(6,1)
             else
               fllks6 = eaa_convey(4,1)
             endif
          endif
       endif
C
C TO WCA-1 BY WPB CANAL
C
         fllkwpb = eaa_convey(7,1)
C
C CALCULATE WATER SUPPLY DELIVERIES FROM LAKE OKEECHOBEE TO WATER
C CONSERVATION AREAS
C
C TO WCA-3A
C
C DEFINE LOGICAL VARIABLE TO REPRESENT CONDITIONS FOR WATER SUPPLY
C BACKPUMPING TO LAKE OKEECHOBEE
C
      IF (STAGELO .LT. ZONECSTG .OR. LWSBPMP) THEN
```

```
        LWSBPMP = .TRUE.
        LWSBPMP = .NOT. (IWSBPCNTR .GE. MAXBPCNTR .OR.
     $    STAGELO .GE. upper_stg_ws_bkp)
      ENDIF
      IF (STAGELO .GE. ZONECSTG ) THEN
         IWSBPCNTR = IWSBPCNTR + 1
      ENDIF
      IF (ZONECSTG .LT. -10.0 .OR. STAGELO .LT. ZONECSTG)
     $  IWSBPCNTR = 0
c     LWSBPMP = .NOT. (IWSBPCNTR .GT. MAXBPCNTR .OR.
c    $    STAGELO .GE. upper_stg_ws_bkp)
C
c
          WSLEC = 0.0
          iserv = iserv_area(is8bsn-6)
          DO j = 1,nsacnl(iserv)
           WSLEC = WSLEC + DQU(isa_canal_indx(iserv,j))
     $          * 86400.
          ENDDO
          if (iconvey_canal_no(is8bsn-6) .ne. 0) then
            iconvey_c = iconvey_canal_no(is8bsn-6)
          else
            iconvey_c = iconvey_canal_no(is150bws-6)
          endif
c
c  if pipeline is used to route water from EAA/LOK to LEC
c
          if (opt_pipeline_ws(is8bsn-6) .eq. 'PIPEL  ') then
             LWSLO_floor(iwca3a) = .TRUE.
             WSWCA3A = 0.0
             rlocal_storage_3a = 0.0
c  if water supply deliveries from WCA1 do not occur daily,then
c  LOK deliveries must be limited to capacity of WCA1 to deliver
c  water when demands in SA1 are high
c
             IF (LFRACBFLR(iwca3a)) THEN
                no_of_days_del_week = no_of_days_wk_wcadel(iwca3a,2)
             ELSE
                no_of_days_del_week = no_of_days_wk_wcadel(iwca3a,1)
             ENDIF
             IF (no_of_days_del_week .LE. 5) THEN

                IWCA = idn_wca_indx(is8bsn-6)
                floor = floor_elev(iwca,iconvey_c)
                do nn = 1,nch
                  do ii = 1,nstrctr(nn)
                    if (flnm(istr(nn,ii)) .eq. 'WSS151') then
                      if (floor .gt.
     $                  chdep(idncnl(nn,ii))) then
                        add_capac_ws = discoef(nn,ii,1,1)
     $                  *(floor-chdep(idncnl(nn,ii)))
     $                  ** pwr(nn,ii,1,1)
                      else
                        add_capac_ws = 0.0
                      endif
                      go to 2223
                    else
```

```
                    add_capac_ws = 0.0
                  endif
 2223             if (use_enp_ws_to_lec .eq.'TRUE ') then
                    if (flnm(istr(nn,ii)) .eq. 'S333  ') then
                      if (floor .gt .chdep(idncnl(nn,ii))) then
                        add_s333_ws = discoef(nn,ii,1,1)
     $                  *(floor-chdep(idncnl(nn,ii)))
     $                  ** pwr(nn,ii,1,1)
                      else
                        add_s333_ws = 0.0
                      endif
                    endif
                    go to 2224
                  else
                    add_s333_ws = 0.0
                  endif
                 enddo
 2224            total_pot_cap = add_capac_ws + add_s333_ws
                enddo
                rlok_pot_del = amin1(total_pot_cap,WSLEC/86400.+FRF)
            ELSE
              rlok_pot_del = WSLEC/86400.+FRF
            ENDIF
            GO TO 100
          endif
C
          if (floor_grid_cell_opt .eq. 'MARSH') then
           do i = 1,no_of_subareas_floor(iwca3a)
            if (avg_sim_wca_stg_floor(iwca3a,i) .lt.
     $           avg_wca_subarea_stg_strt_count(iwca3a,i)) then
              no_days_below_floor(iwca3a,i)
     $           = no_days_below_floor(iwca3a,i) + 1
            else
              no_days_below_floor(iwca3a,i) = 0
            endif
           enddo
           ilok_del_ws = 0
           do i = 1,no_of_subareas_floor(iwca3a)
             if ((no_days_below_floor(iwca3a,i) .gt.
     $         min_consec_days_bel_strtstg .and.
     $         avg_sim_wca_stg_floor(iwca3a,i) .lt.
     $         avg_wca_subarea_crit_stage(iwca3a,i))
     $         .or. (avg_sim_wca_stg_floor(iwca3a,i)
     $         .le. avg_wca_subarea_stg_disc_lok(iwca3a,i)
     $         .and. no_days_deliv_from_lok(iwca3a,i)
     $         .gt. 0) .or. LWSLO_floor_temp(iwca3a,i)) then
               LWSLO_floor_temp(iwca3a,i) = .TRUE.
               LWSLO_floor_temp(iwca3a,i) =
     $            .NOT. (avg_sim_wca_stg_floor(iwca3a,i)
     $            .gt. avg_wca_subarea_stg_disc_lok(iwca3a,i))
             endif
             if (LWSLO_floor_temp(iwca3a,i)) then
               no_days_deliv_from_lok(iwca3a,i) =
     $            no_days_deliv_from_lok(iwca3a,i) + 1
               index_lok_deliv(iwca3a,i) = 1
               ilok_del_ws = ilok_del_ws + 1
             else
```

```
               no_days_deliv_from_lok(iwca3a,i) = 0
               index_lok_deliv(iwca3a,i) = 0
             endif
           enddo
           LWSLO_floor(iwca3a) = ilok_del_ws .gt. 0
          endif
C
        if (iconvey_canal_no(is8bsn-6) .ne. 0) then
          IWCA = idn_wca_indx(is8bsn-6)
          ICA3 = intcnl(idn_wca_indx(is8bsn-6)
     &            ,iconvey_canal_no(is8bsn-6))
          floor = floor_elev(iwca,iconvey_canal_no(is8bsn-6))
          WSWCA3A =(floor - CHDEP(ICA3))*CL(ICA3)*WIDTH(ICA3)
     &    - CGSEEP(ICA3,floor,0,0.)-TOVFLC(ICA3,floor,0,0.,0,1)
c  if water supply deliveries from WCA1 do not occur daily,then
c  LOK deliveries must be limited to capacity of WCA1 to deliver
c  water when demands in SA1 are high
c
          IF (LFRACBFLR(iwca3a)) THEN
             no_of_days_del_week = no_of_days_wk_wcadel(iwca3a,2)
          ELSE
             no_of_days_del_week = no_of_days_wk_wcadel(iwca3a,1)
          ENDIF
          IF (no_of_days_del_week .LE. 5) THEN

             do nn = 1,nch
              do ii = 1,nstrctr(nn)
                if (flnm(istr(nn,ii)) .eq. 'S333  ') then
                  if (use_enp_ws_to_lec .eq.'TRUE ') then
                    if (floor .gt .chdep(idncnl(nn,ii))) then
                       add_s333_ws = discoef(nn,ii,1,1)
     $                  *(floor-chdep(idncnl(nn,ii)))
     $                  ** pwr(nn,ii,1,1)
                    else
                       add_s333_ws = 0.0
                    endif
                  else
                    add_s333_ws = 0.0
                  endif
                  go to 3223
                else
                  add_s333_ws = 0.0
                endif
               enddo
              enddo
 3223         do nn = 1,nch
               do ii = 1,nstrctr(nn)
                if (flnm(istr(nn,ii)) .eq. 'WSS151') then
                  if (floor .gt.
     $                chdep(idncnl(nn,ii))) then
                   add_capac_ws = discoef(nn,ii,1,1)
     $             *(floor-chdep(idncnl(nn,ii)))
     $             ** pwr(nn,ii,1,1)
                  else
                   add_capac_ws = 0.0
                  endif
                  go to 3224
```

```
                else
                  add_capac_ws = 0.0
                endif
               enddo
              enddo
 3224         total_pot_cap = add_capac_ws + add_s333_ws
             rlok_pot_del = amin1(total_pot_cap,WSLEC/86400.+FRF)
           ELSE
             rlok_pot_del = WSLEC/86400.+FRF
           ENDIF
c
          if (LWSLO_floor(iwca3a)) then
            rlkdlws = rlok_pot_del*86400.
           if (nint(RLKDLWS/86400.) .gt. 0) then
             index_ws_frm_outside_source(iwca3a,iconvey_c) = 1
           else
             index_ws_frm_outside_source(iwca3a,iconvey_c) = 0
           endif
           rlok_ws = rlok_pot_del*86400. +  AMIN1(WSWCA3A,0.0)
           rlok_ws = amax1(rlok_ws,0.0)
           if (nint(rlok_ws/86400.) .gt. 0) then
             index_ws_frm_outside_src_canal(iwca3a,iconvey_c) = 1
           else
             index_ws_frm_outside_src_canal(iwca3a,iconvey_c) = 0
           endif
           logic_floor(iwca3a) = 1
          else
           RLKDLWS = rlok_pot_del*86400.+ AMIN1(WSWCA3A,0.0)
           rlocal_storage_3a = AMIN1(WSWCA3A,0.0)
           rlocal_storage_3a = abs(rlocal_storage_3a)/86400.
           RLKDLWS = AMAX1(RLKDLWS,0.0)
           if (nint(RLKDLWS/86400.) .gt. 0) then
             index_ws_frm_outside_src_canal(iwca3a,iconvey_c) = 1
           else
             index_ws_frm_outside_src_canal(iwca3a,iconvey_c) = 0
           endif
           index_ws_frm_outside_source(iwca3a,iconvey_c) = 0
           logic_floor(iwca3a) = 0
          endif
          flow_thru_s8 = (outflow_from_roten_thru_s8
     $                  + dischg_to_s8_frm_sta34)*86400.
          rlocal_storage_3a = rlocal_storage_3a + amin1(flow_thru_s8
     $                          ,rlkdlws)/86400.
          rlkdlws = rlkdlws - amin1(flow_thru_s8,rlkdlws)
          if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa(is7bsn-6)
     $         .eq. 1) then
            rlocal_storage_3a = rlocal_storage_3a + amin1(rlkdlws/86400.,
     $             runoff_thru_s150)
            rlkdlws = rlkdlws - amin1(rlkdlws,runoff_thru_s150*86400.)
          else
            rlocal_storage_3a = rlocal_storage_3a + amin1(rlkdlws/86400.,
     $             kflo(ifls150))
            rlkdlws = rlkdlws - amin1(rlkdlws,kflo(ifls150)*86400.)
          endif
          rlocal_storage_3a = rlocal_storage_3a +
     $     amin1(ws_frm_res_to_wca3a,RLKDLWS/86400.)
          RLKDLWS = RLKDLWS - amin1(ws_frm_res_to_wca3a
```

```
     $          *86400.,RLKDLWS)
         else
          ICA3 = nch+1
          rlok_pot_del = WSLEC/86400.+FRF
          rlkdlws = 0.0
          if (LWSLO_floor(iwca3a)) then
            logic_floor(iwca3a) = 1
          else
            logic_floor(iwca3a) = 0
          endif
         endif
         go to 101
c
c  for pipeline
c
 100     rlkdlws = rlok_pot_del*86400.
         ICA3 = nch+1
         if (LWSLO_floor(iwca3a)) then
           logic_floor(iwca3a) = 1
         else
           logic_floor(iwca3a) = 0
         endif
         flow_thru_s8 = (outflow_from_roten_thru_s8
     $                   + dischg_to_s8_frm_sta34)*86400.
         ws_pipeline_3a = ws_pipeline_3a + amin1(flow_thru_s8,rlkdlws)
     $                              / 86400.
         rlocal_storage_3a = rlocal_storage_3a + amin1(flow_thru_s8,
     $               rlkdlws)/86400.
         rlkdlws = rlkdlws - amin1(flow_thru_s8,rlkdlws)
         if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa(is7bsn-6)
     $         .eq. 1) then
           ws_pipeline_3a_s150 = ws_pipeline_3a_s150
     $         + amin1(rlkdlws/86400.-frf,runoff_thru_s150)
           rlocal_storage_3a=rlocal_storage_3a +amin1(rlkdlws-frf*86400.
     $                 ,runoff_thru_s150*86400.) / 86400.
           rlkdlws = rlkdlws - amin1(rlkdlws-frf*86400.
     $                ,runoff_thru_s150*86400.)
         else
           ws_pipeline_3a_s150 = ws_pipeline_3a_s150
     $         + amin1(rlkdlws/86400.,kflo(ifls150))
           rlocal_storage_3a=rlocal_storage_3a + amin1(rlkdlws/86400.
     $         ,kflo(ifls150))
           rlkdlws = rlkdlws - amin1(rlkdlws,kflo(ifls150)*86400.)
         endif
         ws_pipeline_3a = ws_pipeline_3a + amin1(ws_frm_res_to_wca3a
     $          ,RLKDLWS/86400.)
         rlocal_storage_3a=rlocal_storage_3a
     $      + amin1(ws_frm_res_to_wca3a,RLKDLWS/86400.)
         RLKDLWS = RLKDLWS - amin1(ws_frm_res_to_wca3a
     $          *86400.,RLKDLWS)
 101    dmnd_lec_sa3 = rlkdlws/86400.
c
c  determine SA3 demands for S150 if Miami Canal is not used
c  to meet SA3 demands
c
         IWCA_S150 = idn_wca_indx(is150bws-6)
         if ((iconvey_canal_no(is150bws-6) .ne.
```

```
     &     iconvey_canal_no(is8bsn-6) .and.
     &     idn_wca_indx(is150bws-6) .eq. idn_wca_indx(is8bsn-6)
     &      .and. iconvey_canal_no(is150bws-6) .ne. 0)
     &      .or. opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
c
c  if pipeline is used to route water from EAA/LOK to LEC
c
          if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
             LWSLO_floor(IWCA_S150) = .TRUE.
             WSWCA3A = 0.0
             GO TO 201
          endif
          ICA3_S150 = intcnl(idn_wca_indx(is150bws-6)
     &            ,iconvey_canal_no(is150bws-6))
c
          iconvey_c = iconvey_canal_no(is150bws-6)
          floor = floor_elev(iwca_s150,iconvey_canal_no(is150bws-6))
          WSWCA3A =(floor - CHDEP(ICA3_S150))*CL(ICA3_S150)
     &    *WIDTH(ICA3_S150)- CGSEEP(ICA3_S150,floor,0,0.)
     &    -TOVFLC(ICA3_S150,floor,0,0.,0,1)
c
         if (LWSLO_floor(IWCA_S150)) then
           rlkdlws_s150 = rlok_pot_del*86400.
           rlocal_storage_3a = 0.0
          if (nint(RLKDLWS_S150/86400.) .gt. 0) then
            index_ws_frm_outside_source(IWCA_S150,iconvey_c) = 1
          else
            index_ws_frm_outside_source(IWCA_S150,iconvey_c) = 0
          endif
          rlok_ws_s150 = rlok_pot_del*86400.+ AMIN1(WSWCA3A,0.0)
          rlok_ws_s150 = amax1(rlok_ws_s150,0.0)
          if (nint(rlok_ws_s150/86400.) .gt. 0) then
            index_ws_frm_outside_src_canal(IWCA_S150,iconvey_c) = 1
          else
            index_ws_frm_outside_src_canal(IWCA_S150,iconvey_c) = 0
          endif
          logic_floor(IWCA_S150) = 1
         else
          RLKDLWS_S150 = rlok_pot_del*86400.+ AMIN1(WSWCA3A,0.0)
          RLKDLWS_S150 = AMAX1(RLKDLWS_S150,0.0)
          rlocal_storage_3a = AMIN1(WSWCA3A,0.0)
          rlocal_storage_3a = abs(rlocal_storage_3a)/86400.
          if (nint(RLKDLWS_S150/86400.) .gt. 0) then
            index_ws_frm_outside_src_canal(IWCA_S150,iconvey_c) = 1
          else
            index_ws_frm_outside_src_canal(IWCA_S150,iconvey_c) = 0
          endif
          index_ws_frm_outside_source(IWCA_S150,iconvey_c) = 0
          logic_floor(IWCA_S150) = 0
         endif
         if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa(is7bsn-6)
     $         .eq. 1) then
           rlocal_storage_3a = rlocal_storage_3a + amin1(rlkdlws_s150
     $                ,runoff_thru_s150*86400.) / 86400.
           rlkdlws_s150 = rlkdlws_s150 - amin1(rlkdlws_s150
     $                ,runoff_thru_s150*86400.)
         else
```

```
          rlocal_storage_3a = rlocal_storage_3a + amin1(rlkdlws_s150
     $                ,kflo(ifls150)*86400.)/86400.
          rlkdlws_s150 = rlkdlws_s150 - amin1(rlkdlws_s150
     $                ,kflo(ifls150)*86400.)
        endif
        rlocal_storage_3a = rlocal_storage_3a
     $     + amin1(ws_frm_res_to_wca3a*86400.,RLKDLWS_S150)
     $     / 86400
        RLKDLWS_S150 = RLKDLWS_S150 - amin1(ws_frm_res_to_wca3a
     $          *86400.,RLKDLWS_S150)
        else
         rlkdlws_s150 = 0.0
         if (LWSLO_floor(IWCA_S150)) then
           logic_floor(IWCA_S150) = 1
         else
           logic_floor(IWCA_S150) = 0
         endif
        endif
        go to 202
c
c  for pipeline
c
 201    rlkdlws_s150 = rlok_pot_del*86400.
        logic_floor(IWCA_S150) = 1
        if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa(is7bsn-6)
     $         .eq. 1) then
          ws_pipeline_3a_s150 = ws_pipeline_3a_s150
     $            + amin1(rlkdlws_s150/86400.,runoff_thru_s150)
          rlocal_storage_3a = rlocal_storage_3a +
     $            amin1(rlkdlws_s150/86400.,runoff_thru_s150)
          rlkdlws_s150 = rlkdlws_s150 - amin1(rlkdlws_s150
     $                ,runoff_thru_s150*86400.)
        else
          ws_pipeline_3a_s150 = ws_pipeline_3a_s150
     $            + amin1(rlkdlws_s150/86400.,kflo(ifls150))
          rlocal_storage_3a = rlocal_storage_3a
     $            + amin1(rlkdlws_s150/86400.,kflo(ifls150))
          rlkdlws_s150 = rlkdlws_s150 - amin1(rlkdlws_s150
     $                ,kflo(ifls150)*86400.)
        endif
        ws_pipeline_3a_s150 = ws_pipeline_3a_s150 +
     $          amin1(ws_frm_res_to_wca3a,RLKDLWS_S150/86400.)
        rlocal_storage_3a = rlocal_storage_3a +
     $          amin1(ws_frm_res_to_wca3a,RLKDLWS_S150/86400.)
        RLKDLWS_S150 = RLKDLWS_S150 - amin1(ws_frm_res_to_wca3a
     $          *86400.,RLKDLWS_S150)
  202   dmnd_lec_sa3_s150 = rlkdlws_s150/86400.
        if (ifacteaa .eq. 1) then
          grf3amx_prescap = grf3amx/amax1(rcpfacteaa_canal(is8bsn-6)
     $                ,1.0)
        endif
c
c  compute runoff from 298 Districts in EAA diverted to STAs
c
        if (nmarea .gt. 0) then
          tmpvar = 0.0
          call flow_to_stas(runoff_298_districts,0.0,tmpvar
```

```
     $     ,4,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $     ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $     ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $     ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $     ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $     ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $     ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $     ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $     ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
         endif
         runoff_south_bef_divers(is8bsn-6) = runoff_south(is8bsn-6)
c
c  determine if criteria are met for environmental ws releases
c   from LOK to WCA-3A
c
         ntarg = 0
         if (lrunoff_to_res_when_above_targ .eq. 'FALSE') then
           do itarg = 1,no_targ_loc(is8bsn-6)
             if (LWSLO(itarg_area(itarg,is8bsn-6))) then
               ntarg = ntarg + 1
             endif
           enddo
         else
           do itarg = 1,no_targ_loc(is8bsn-6)
             if (avg_sim_wca_stg(itarg_area(itarg,is8bsn-6))
     $         .lt. stage_import_wca(itarg_area(itarg,is8bsn-6))
     $         + offset(itarg_area(itarg,is8bsn-6),1)) then
               ntarg = ntarg + 1
             endif
           enddo
         endif
C
c
c  route eaa runoff to asr reservoir if there are no demands in LEC
c
         IF (wcaenv(idn_wca_indx_env(is8bsn-6)) .eq. 'FALSE') then
          if (iasr_opt_eaa(is8bsn-6) .eq. 1 .or.
     $         ires_opt_eaa(is8bsn-6) .eq. 1) then
           runoff_to_inject = amax1(runoff_south(is8bsn-6)
     $          - dmnd_lec_sa3-wsuply(indxsta(is8bsn-6))+frf,0.0)
           flow_to_eaa_res_rnff(is8bsn-6) = amin1(runoff_to_inject
     $     *frac_runoff_asr(is8bsn-6),volwtr(ieaa_res_asr(is8bsn-6))
     $     ,weircff(ieaa_res_asr(is8bsn-6),1,1))
           rem_weir_capac(is8bsn-6) =weircff(ieaa_res_asr(is8bsn-6),1,1)
     $          - flow_to_eaa_res_rnff(is8bsn-6)
           volwtr(ieaa_res_asr(is8bsn-6))=volwtr(ieaa_res_asr(is8bsn-6))
     $           - flow_to_eaa_res_rnff(is8bsn-6)
           runoff_south(is8bsn-6) = runoff_south(is8bsn-6)
     $           - flow_to_eaa_res_rnff(is8bsn-6)
           if (istaopt(is8bsn-6) .eq. 0) then
              q = runoff_south(is8bsn-6) + flow_thru_s8
              IF(Q .GT. S8MAX) THEN
                KFLO(IFLS8) = S8MAX
                IAGFLO(IAG(IS8BSN),1) = NINT(Q-S8MAX) + IAGFLO(IAG
     $            (IS8BSN),1)
              ELSE
                KFLO(IFLS8) = Q
```

```
              ENDIF
            endif
           endif
          ELSE
c
c route eaa runoff to asr reservoir if there are no LEC demands and
c no environmental demands in everglades
c
            if (iasr_opt_eaa(is8bsn-6) .eq. 1 .or.
     $          ires_opt_eaa(is8bsn-6) .eq. 1) then
c             if (.not.(lwslo(iwca3a).or.lwslo(inwcwca3a))) then
              if (ntarg .eq. 0) then
                runoff_to_inject = amax1(runoff_south(is8bsn-6)
     $               - dmnd_lec_sa3-wsuply(indxsta(is8bsn-6))+frf,0.0)
              else
                rmax_flow_thru = amin1(s354mx - amax1(kflo(ifls3),0.0)
     $              -wsuply(indxsta(is8bsn-6)),grf3amx
     $              -amax1(kflo(ifls3),0.0)-wsuply(indxsta(is8bsn-6))
     $              ,s8max)
                rmax_flow_thru = amax1(rmax_flow_thru,0.0)
                rmax_flow_bypass_res =amax1(rmax_flow_thru,dmnd_lec_sa3
     $                  - frf)
                runoff_to_inject = amax1(runoff_south(is8bsn-6)
     $                      - rmax_flow_bypass_res,0.0)
                if (divers_excess_to_res .eq. 'FALSE') then
                  runoff_to_inject = 0.0
                endif
              endif
              flow_to_eaa_res_rnff(is8bsn-6) = amin1(runoff_to_inject
     $        *frac_runoff_asr(is8bsn-6),volwtr(ieaa_res_asr(is8bsn-6))
     $        ,weircff(ieaa_res_asr(is8bsn-6),1,1))
              rem_weir_capac(is8bsn-6) =weircff(ieaa_res_asr(is8bsn-6)
     $          ,1,1) - flow_to_eaa_res_rnff(is8bsn-6)
              volwtr(ieaa_res_asr(is8bsn-6))
     $        = volwtr(ieaa_res_asr(is8bsn-6))-flow_to_eaa_res_rnff
     $           (is8bsn-6)
              runoff_south(is8bsn-6) = runoff_south(is8bsn-6)
     $          - flow_to_eaa_res_rnff(is8bsn-6)
              if (istaopt(is8bsn-6) .eq. 0) then
                q = runoff_south(is8bsn-6) + flow_thru_s8
                IF(Q .GT. S8MAX) THEN
                  KFLO(IFLS8) = S8MAX
                  IAGFLO(IAG(IS8BSN),1) = NINT(Q-S8MAX) + IAGFLO(IAG
     $              (IS8BSN),1)
                ELSE
                  KFLO(IFLS8) = Q
                ENDIF
              endif
            endif
          ENDIF
c
          ntarg = 0
          do itarg = 1,no_targ_loc(is8bsn-6)
            if (LWSLO(itarg_area(itarg,is8bsn-6))) then
              ntarg = ntarg + 1
            endif
          enddo
```

```
c
      IF(KFLO(IFLS3).GE.0.) THEN
C
C NO FLOOD CONTROL BACKPUMPING FROM EAA TO LOK EXISTS, THUS WATER SUPPLY
C DELIVERY FROM LOK TO WCA AND LEC IS POSSIBLE, AND WATER SUPPLY
C BACKPUMPING MAY BE POSSIBLE
C
         if (iconv_use_inj(is8bsn-6) .eq. 0) then
           KFLOREF(is8bsn-6) = KFLO(IFLS3)+runoff_south(is8bsn-6)
         else
           KFLOREF(is8bsn-6) = KFLO(IFLS3) +
     $                runoff_south_bef_divers(is8bsn-6)
         endif
         add_conv_red_miami = 0.0
         do irs = 1,nres_ws
          if (iconv_use_opt_ws(irs) .eq. 1) then
            do js = 1,noutlets(irs)
              if (ieaa_bsn(irs,js) .eq. is8bsn-6) then
                kfloref(is8bsn-6) = kfloref(is8bsn-6)
     $                  + res_outflow_to_basin(irs,js)
                add_conv_red_miami = add_conv_red_miami
     $                  + res_outflow_to_basin(irs,js)
              endif
            enddo
          endif
         enddo
C COMPUTE INITIAL CONVEYANCE CAPACITY
         IF(GRF3AMX.GT.0.0) THEN
            GRFLK3A = AMAX1(GRF3AMX - KFLOREF(is8bsn-6),0.0)
C NOTE: CONVEYANCE CAPACITY WILL BE REDUCED BY THE AMOUNT OF SOUTHWARD FLOW
C       THROUGH S3 THAT WILL BE USED TO SATISFY DEMAND IN THE EAA OR THROUGH
C       S8 TO REMOVE RUNOFF FROM THE EAA
         ELSE
            GRFLK3A = 0.0
         ENDIF
         if (istaopt(is8bsn-6) .eq. 0) then
          if (eaa_conv_opt_s8 .eq. 'PUMP') then
           if (((dmnd_lec_sa3 .gt. grf3amx .or. kflo(ifls8) .gt.
     $         grf3amx) .and. grf3amx .lt. grf3amx_pump
     $         .and. s354mx .lt. s354cap_pump) .or. dmnd_lec_sa3
     $         .gt. s8_cap) THEN
              grflk3a = amax1(grf3amx_pump - kfloref(is8bsn-6),0.0)
              igravity_s8_nosta = 0
              if (s354mx .lt. s354cap_pump) then
                s354mx  = s354cap_pump
              endif
              s354_tw = s354_tw_pump
              miamic_cap_upper_grav = grf3amx_pump
           else
            igravity_s8_nosta = 1
           endif
          else
            igravity_s8_nosta = 1
          endif
         endif
         dmnd_miami_basin = kflo(ifls3) - water_supply_to_holyl
c
```

```
c  deliver water from Lake OKeechobee to maintain STA34 via Miami Canal
c  during dry periods
c
         if (istaopt(is8bsn-6) .eq. 1) then
          ireach = 5
          ctw = pmp_int_new(is8bsn-6)
          if (.not. LFORPUMP(is8bsn-6)) then
            istype = 1
            istr_eaa = 5
            chw = stagelo
            delta0 = (chw - ctw) / 25.0
            call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo
     $           ,delta0,1.0,100,miamic_cap_wsdeltosta,str_cap,str_tw
     $           ,ifacteaa)
            if (str_tw .gt. rmaxstg(is8bsn-6)) then
              str_tw = rmaxstg(is8bsn-6)
              str_cap = -1.0
              call eaaspill(str_cap,stagelo,rmaxstg(is8bsn-6),5,
     $        ifacteaa)
              qcanal = -1.0
              if (str_tw.gt.ctw) then
                call eaa_prof_lu_table(qcanal,str_tw,ctw,
     $           ireach,ifacteaa)
              else
                qcanal = 0.0
              endif
              miamic_cap_wsdeltosta = qcanal
            endif
          else
            istr_eaa = 9
            str_tw=amin1(stagelo+rmax_lift(is8bsn-6),rmaxstg(is8bsn-6))
            str_cap = -1.0
            call eaaspill(str_cap,stagelo,str_tw,istr_eaa,ifacteaa)
            qcanal = str_cap
            str_tw_max = str_tw
            str_tw = -1.0
            call eaa_prof_lu_table(qcanal,str_tw,ctw,ireach,ifacteaa)
            if (str_tw .gt. str_tw_max) then
              str_tw = str_tw_max
              qcanal = -1.0
              call eaa_prof_lu_table(qcanal,str_tw,ctw,ireach,ifacteaa)
            endif
            miamic_cap_wsdeltosta = qcanal
          endif
          rem_cap_wstosta = miamic_cap_wsdeltosta - kfloref(is8bsn-6)
          rem_cap_wstosta = amax1(rem_cap_wstosta,0.0)
          rem_cap_struc = amax1(str_cap - kflo(ifls3),0.0)
          wsdel_to_sta(indxsta(is8bsn-6)) = amin1(wsuply(indxsta
     $                (is8bsn-6)),rem_cap_wstosta,rem_cap_struc)
          grflk3a = grflk3a - amin1(wsdel_to_sta(indxsta(is8bsn-6))
     $               ,grflk3a)
          kflo(ifls3) = kflo(ifls3) + wsdel_to_sta(indxsta(is8bsn-6))
          wss354 = wss354 + wsdel_to_sta(indxsta(is8bsn-6))
          kfloref(is8bsn-6) = kfloref(is8bsn-6)
     $               + wsdel_to_sta(indxsta(is8bsn-6))
          wsuply(indxsta(is8bsn-6)) = wsuply(indxsta(is8bsn-6))
     $       - wsdel_to_sta(indxsta(is8bsn-6))
```

```
          ws_to_sta34_frm_lok = ws_to_sta34_frm_lok
     $        + wsdel_to_sta(indxsta(is8bsn-6))
         endif
C
C ROUTE WATER FROM LOK TO SEMINOLE INDIANS AND HOLEYLAND VIA MIAMI CANAL
C
         GWDPTH = (ELLS(NODEHL) - H(NODEHL)) * S(NODEHL)
         IF (POND(NODEHL) .GT. GWDPTH) THEN
           HLSTAGE = POND(NODEHL) - GWDPTH + ELLS(NODEHL)
         ELSE
           HLSTAGE = (POND(NODEHL) - GWDPTH)/S(NODEHL) + ELLS(NODEHL)
         ENDIF
         call bpts(iday,month,10.0,7,1,1,holey_sched)
         IF (MONTH .EQ. 1) THEN
           PREV = RSIAPM(12)
         ELSE
           PREV = RSIAPM(MONTH - 1)
         ENDIF
         RSIAP = PREV + IDAY*(RSIAPM(MONTH) - PREV)/NUMD(MONTH)
         if (hlstage .lt. holey_sched .and. ws_to_holy_opt .ne.'BRUSHF'
     $    .and. stagelo .gt. RSIAP) then
            rmax_ws_from_lok_to_holy = rem_inflow_to_meet_holy_sched
         else
            rmax_ws_from_lok_to_holy = 0.0
         endif
         if (isem_flg .eq. 1) then
           dmnd_sugh = dmnd_sugh_orig
           sem_dmnd_met_bef_lok = sem_dmnd_orig - sem_cyp_dmnd(jday)
           rem_cap_for_sply_sem = capac_semws-(sem_dmnd_orig/1.9835
     $          - sem_cyp_dmnd(jday)/1.9835)
           rem_cap_for_sply_sem = amax1(rem_cap_for_sply_sem,0.0)
           rem_cap_new_pump = capc_new_pump_to_west_frm_roten
     $         - (outflow_from_roten_to_west+sta34_out_to_wca3anw
     $         + sta34_out_to_s140+holy_out_to_s140+holey_out_to_nwc)
c          print *,rem_cap_new_pump,capc_new_pump_to_west_frm_roten
c    $     ,outflow_from_roten_to_west,sta34_out_to_wca3anw
c    $     ,sta34_out_to_s140,holy_out_to_s140,holey_out_to_nwc
c    $     ,'   BIGCYP  ',sem_cyp_dmnd(jday)
c           rem_cap_for_sply_sugh = capac_sugh-(dmnd_sugh_orig-dmnd_sugh)
c     $          /1.9835
           if (issm_cutb_opt_bcyp .eq. 1) then
              rem_dmnd_sem_cyp = sem_cyp_dmnd(jday)*pctmet/1.9835
           else
              rem_dmnd_sem_cyp = sem_cyp_dmnd(jday)/1.9835
           endif
c           rem_dmnd_sugh    = dmnd_sugh/1.9835*pctmet
           rem_cap_for_sply_sem = amin1(rem_cap_for_sply_sem
     $         ,rem_dmnd_sem_cyp,rem_cap_new_pump)
c           rem_cap_for_sply_sugh = amin1(rem_cap_for_sply_sugh
c     $         ,rem_dmnd_sugh)
           total_rem_sply = rem_cap_for_sply_sem
     $          + rmax_ws_from_lok_to_holy
c     $         + rem_cap_for_sply_sugh
           rem_cap_s3  = max(s354mx - kflo(ifls3),0.0)
           rem_cap_miami = amin1(rem_cap_s3,GRFLK3A)
           if (total_rem_sply .gt. 0.0) then
             ratio_ws = amin1(rem_cap_miami/total_rem_sply,1.0)
```

```
            else
              ratio_ws = 1.0
            endif
c           print *,rem_cap_s3,s354mx,kflo(ifls3),GRFLK3A
c    $        ,rem_cap_for_sply_sem,rmax_ws_from_lok_to_holy
c    $        ,rem_cap_miami,total_rem_sply,ratio_ws,' KLJL'
            ws_deliv = ratio_ws * total_rem_sply
            kflo(ifls3) = kflo(ifls3) + ws_deliv
            kfloref(is8bsn-6) = kfloref(is8bsn-6) + ws_deliv
            GRFLK3A = GRFLK3A - ws_deliv
            ws_frm_lok_to_semcyp = ratio_ws*rem_cap_for_sply_sem
c            ws_frm_lok_to_sugh   = ratio_ws*rem_cap_for_sply_sugh
            ws_frm_lok_to_holy   = ratio_ws*rmax_ws_from_lok_to_holy
            dellec = dellec + ws_frm_lok_to_holy * 1.9835
            dellec = dellec + ws_frm_lok_to_semcyp*1.9835
c            dmnd_miami_basin = dmnd_miami_basin + ws_frm_lok_to_sugh
            wss354 = wss354 + ws_frm_lok_to_semcyp + ws_frm_lok_to_holy
            water_supply_to_holyl = ws_frm_lok_to_holy
     $           + water_supply_to_holyl
c            iagflo(iag(isugbsn),1) = iagflo(iag(isugbsn),1) -
c     $         amin1(dmnd_sugh/1.9835,capac_sugh) + ws_frm_lok_to_sugh
c           if (ifyr_iyear-1 .eq. 1970 .and. month .ge. 9)
c    $      print *,dmnd_sugh/1.9835,sem_cyp_dmnd(jday)/1.9835,pctmet,
c    $       ratio_ws,iagflo(iag(isugbsn),1),total_rem_sply,ws_deliv
c    $       ,rem_cap_for_sply_sem,rem_cap_for_sply_sugh
c    $       ,rem_cap_miami,month,iday,capac_semws,capac_sugh
            sem_cyp_dmnd(jday) = sem_cyp_dmnd(jday)
     $         - ws_frm_lok_to_semcyp*1.9835
          endif
C
c
          IF (wcaenv(idn_wca_indx_env(is8bsn-6)) .eq. 'FALSE') then
C CRMIN() IN CA3 CANAL USED AS TRIGGER
C MAKE NECESSARY DELIVERY BASED ON CALCULATED TOTAL WCA AND LEC NEEDS
C23456789*123456789*123456789*123456789*123456789*123456789*123456789*12
C
            if (istaopt(is8bsn-6) .eq. 1 .and. iwsbkpwsta .eq. 1) then
C
              IF(nint(RLKDLWS/86400.).GE.0) THEN
                RLKDLCFS = RLKDLWS/86400.
                IF(runoff_south(is8bsn-6) .GT. RLKDLCFS) THEN
                   IF(LWSBPMP.AND.KFLO(IFLS3).GE.0.) THEN
                     IF (LWSBKP_floor(is8bsn-6) .or. LWSLO_floor(iwca3a))
     $                THEN
                       ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is8bsn-6)
                     ELSE
                       ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is8bsn-6)
                     ENDIF
                     ws_bkp_frac_runff = 1.0
                     KFLO(IFLS3) = - amin1((runoff_south(is8bsn-6) -
     $                RLKDLCFS)*ws_bkp_frac *ws_bkp_frac_runff,
     $                rmax_capac_ws_bkp(is8bsn-6))
                      runoff_south(is8bsn-6) = runoff_south(is8bsn-6) -
     $                amin1((runoff_south(is8bsn-6)-RLKDLCFS)*ws_bkp_frac
     $                *ws_bkp_frac_runff,rmax_capac_ws_bkp(is8bsn-6))
                   ENDIF
                ENDIF
```

```
              RLKDLWS = RLKDLCFS*86400.
            ENDIF
          endif
         if (istaopt(is8bsn-6) .eq. 1) then
            runoff_s8bsn = runoff_south(is8bsn-6)
            tmpvar = gflk3ad
            call flow_to_stas (runoff_s8bsn,rlkdlws/86400.
     $    ,tmpvar,1,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $    ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $    ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $    ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $    ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $    ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $    ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $    ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $    ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
          else
            flow_from_eaa_lok_s8 = kflo(ifls8)
     $           - outflow_from_roten_thru_s8
          endif
          if (iwsbkpwsta .eq. 1.and.istaopt(is8bsn-6) .eq. 1) RLKDLWS =
     $     RLKDLWS - amin1(RLKDLWS ,runoff_south(is8bsn-6)*86400.)

C
C DETERMINE THE AMOUNT OF RUNOFF IN MIAMI CANAL BASIN IN THE EAA TO BE
C BACKPUMPED TO LAKE OKEECHOBEE FOR WATER SUPPLY.  WILL BE DONE ONLY
C IF LAKE OKEECHOBEE LEVEL IS LOW AND THERE IS NO REMAINING WS OR ENVIRONMENTAL
C NEEDS IN THE WCA'S OR LOWER EAST COAST AFTER EAA RO IS USED TO SATISFY
C ORIGINAL WS OR ENVIRONMENTAL NEEDS IN WCA AND LEC
            IF (LWSBPMP.AND.KFLO(IFLS3).GE.0..AND.(iwsbkpwsta.eq.0
     $         .or. istaopt(is8bsn-6) .eq. 0)) THEN
              IF (LWSBKP_floor(is8bsn-6) .or. LWSLO_floor(iwca3a)) THEN
                 ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is8bsn-6)
              ELSE
                 ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is8bsn-6)
              ENDIF
              ws_bkp_frac_runff = 1.0
            ENDIF
            IF (iwsbkpwsta .eq. 0 .or. istaopt(is8bsn-6) .eq. 0) THEN
             IF(nint(dmnd_lec_sa3) .GT. 0) THEN
C WITH WCA AND LEC DEMAND
               RLKDLCFS = rlkdlws/86400.
c
                IF( flow_from_eaa_lok_s8 .GT. RLKDLCFS) THEN
                  IF(LWSBPMP.AND.KFLO(IFLS3).EQ.0.) THEN
C WS BP IS DESIRED AND POSSIBLE SINCE S3 IS NOT YET UTILIZED AT THIS POINT;
C REDISTRIBUTE FLOWS BETWEEN S8 AND S3 BY CONSIDERING RLKDLWS
C PARTITION S8 FLOW TO SOUTH (I.E. WHAT GOES TO MEET ALL WCA AND LEC DEMAND)
C AND TO NORTH (I.E. WHAT GOES AS WATER SUPPLY BACKPUMPING)
                     KFLO(IFLS3)=KFLO(IFLS3)-amin1((flow_from_eaa_lok_s8
     $                    - RLKDLCFS)*ws_bkp_frac*ws_bkp_frac_runff
     $                    ,rmax_capac_ws_bkp(is8bsn-6))
                     KFLO(IFLS8) = kflo(ifls8)
     $                 - amin1((flow_from_eaa_lok_s8 - RLKDLCFS)
     $                 *ws_bkp_frac*ws_bkp_frac_runff
     $                 ,rmax_capac_ws_bkp(is8bsn-6))
                  ENDIF
```

```
                 RLKDLCFS = 0.0
              ELSE
C S8 FLOW CANNOT MEET TOTAL WCA AND LEC REQUIREMENT; NO WS BACKPUMPING POSSIBLE;
C JUST REDUCE WS NEED BASED ON S8 CONTRIBUTION
                 RLKDLCFS = RLKDLCFS - flow_from_eaa_lok_s8
              ENDIF
              RLKDLWS = RLKDLCFS*86400.
            ELSE
C WITHOUT WCA AND LEC DEMAND
              IF(LWSBPMP.AND.KFLO(IFLS3).EQ.0) THEN
C REDISTRIBUTE FLOWS BETWEEN S8 AND S3
C ALL EAA RO WILL BE BACKPUMPED TO LOK; REDISTRIBUTE S3 AND S8 FLOWS
C ANY ADDITIONAL CONSTRAINTS HERE ?
                 KFLO(IFLS3) = KFLO(IFLS3) - amin1(flow_from_eaa_lok_s8
     $                *ws_bkp_frac*ws_bkp_frac_runff
     $                ,rmax_capac_ws_bkp(is8bsn-6))
                 KFLO(IFLS8) = kflo(ifls8)
     $                - amin1(flow_from_eaa_lok_s8
     $                *ws_bkp_frac*ws_bkp_frac_runff
     $                ,rmax_capac_ws_bkp(is8bsn-6))
              ENDIF
            ENDIF
          ENDIF
C CALCULATE TOTAL REQUIREMENT AND MAKE LOK DELIVERY IF TOTAL WCA AND LEC
C REQUIREMENT IS POSITIVE (AFTER S8 CONTRIBUTION HAS BEEN CONSIDERED)
          IF(nint(RLKDLWS/86400.) .GT.0) THEN
C LOK CONTRIBUTION TO MEET WCA+LEC DEMAND STILL NEEDED
              LWSWCA = .TRUE.
c
              if (runoff_south(is8bsn-6) .le. dmnd_lec_sa3
     $            .or. istaopt(is8bsn-6) .eq. 0) then
                WSDOWN = RLKDLWS/86400.
                avl_cap = s354mx - amax1(kflo(ifls3),0.0)
                avl_cap = amax1(avl_cap,0.0)
                GFLK3AD = AMIN1(GRFLK3A,WSDOWN,avl_cap)
                if (iconvey_c .ne. 0) then
                 if (nint(wsdown) .gt. 0) then
                  lec_dmnd_on_lok(iwca3a,iconvey_c) = 1
                 else
                  lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
                 endif
                endif
              else
                GFLK3AD = 0.0
                if (iconvey_c .ne. 0) then
                  lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
                endif
              endif
          ELSE
              LWSWCA = .FALSE.
              WSDOWN = 0.0
              GFLK3AD = 0.0
              if (iconvey_c .ne. 0) then
                lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
              endif
          ENDIF
c
```

```
            if (IPRORTE(IWCA3A) .EQ. 1) THEN
              if (nint(dmnd_lec_sa3) .gt.0) then
                prorate_ratio = gflk3ad / (dmnd_lec_sa3
     $                + rlocal_storage_3a)
                prorate_ratio = amin1(prorate_ratio,1.0)
              else
                prorate_ratio = 0.0
              endif
              rlok_flow_to_enp_s354 = prorate_ratio * frf
            endif
          flow_through_ws = gflk3ad
c
          do irs = 1,nres_ws
           do js = 1,noutlets(irs)
             is = istrctr_ws(ires_no(irs),js)
             if (idshg_opt(ires_no(irs),js) .eq. 3) then
              IF (resname(iwmads(ires_no(irs),is)).eq.
     $        name_res_spec_code(1)) THEN
c             if (res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(19) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(20) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(21) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(22) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(23)) then
               if (ifacteaa .eq. 1) then
                 flow_thru_prescap = amax1(grf3amx_prescap - kfloref
     $                 (is8bsn-6) ,0.0)
                 flow_thru_prescap = amin1(flow_thru_prescap,gflk3ad)
                 total_inflow_prescap = runoff_south(is8bsn-6)
     $              + flow_thru_prescap
               else
                 total_inflow_prescap = runoff_south(is8bsn-6)
     $              + gflk3ad
               endif
               if (total_inflow_prescap
     $             .gt. dmnd_lec_sa3) then
                 runoff_to_sta = runoff_south(is8bsn-6)
               else
                 runoff_to_sta = 0.0
               endif
              ELSEIF (resname(iwmads(ires_no(irs),is)).eq.
     &        name_res_spec_code(4)) THEN
                runoff_to_sta =inflow_to_sta6_frm_unit1
              ELSE
                runoff_to_sta = 0.0
              ENDIF
              call ws_from_res(irs,js,ires_no(irs),ieaa_bsn(irs,js)
     $        ,crmin,sumrain,jday,isem_flg,dmnd_sugh,dmnd_sugh_orig
     $        ,capac_semws,capac_sugh,stgref_asr_rec
     $        ,res_outflow_to_basin,stagelo_beg,pctmet
     $        ,sem_dmnd_orig,0,outflow_from_sta2_for_ws
     $        ,wsndes,wsndese,runoff_to_sta,volwtr
     $        ,frac_thru_s8_sta34,wsnde1,wsnde2,wsnde3,res_outflow_env
```

```
     $         ,rloc_excess_for_ws,tot_flw_to_c9r_frm_nlkblt
     $         ,ws_from_acmres_lwdd,issm_cutb_opt_bcyp
     $         ,pot_del_to_eaa_basin,alocnnr_potdel
     $         ,total_lec_dmnd_lok,iroute_to_wca3a_frm_holy
     $         ,iopt_env_eaar_thru_g372)
             endif
           enddo
          enddo
         ELSE
C 8.5 FT 3-GAGE AVERAGE USED AS TRIGGER
C MAKE NECESSARY DELIVERY BASED ON 3-GAGE AVERAGE TRIGGER VIOLATION
C NOTE: WSDOWN IS DEFINED ONLY WHEN TRIGGER IS BASED ON
C       CANAL STAGE, I.E., CRMIN()
            avl_cap     = s354mx - amax1(kflo(ifls3),0.0)
            avl_cap = amax1(avl_cap,0.0)
            IF (ntarg .gt. 0) THEN
C STAGE IN WCA-3A IS LOW; WCA-3A NEEDS WS RELEASE FROM LOK
C MAKE MAX. WS DELIVERY BASED ON CONVEYANCE CAPACITY ADJUSTED FOR EAA RO OR DMD
               LWSWCA = .TRUE.
               GFLK3AD = AMIN1(GRFLK3A,avl_cap)
               if (iconvey_c .ne. 0) then
                if (runoff_south(is8bsn-6) .ge. dmnd_lec_sa3) then
                  lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
                else
                  lec_dmnd_on_lok(iwca3a,iconvey_c) = 1
                endif
               endif
            ELSE
               RLKDLWS  = RLKDLWS  - runoff_south(is8bsn-6)*86400.
               WSDOWN = AMAX1(RLKDLWS/86400.,0.0)
               if (nint(WSDOWN) .eq. 0) then
                 LWSWCA = .FALSE.
                 if (iconvey_c .ne. 0) then
                   lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
                 endif
               else
                 LWSWCA = .TRUE.
                 if (iconvey_c .ne. 0) then
                   lec_dmnd_on_lok(iwca3a,iconvey_c) = 1
                 endif
               endif
               GFLK3AD = AMIN1(GRFLK3A,WSDOWN,avl_cap)
            ENDIF
c           print *,GFLK3AD,LWSWCA,GRFLK3A,avl_cap,'  MNMNMN   '
c    $      ,s354mx,kflo(ifls3),ntarg
            flow_through_ws = gflk3ad
            if (IPRORTE(IWCA3A) .EQ. 1) THEN
              if (nint(dmnd_lec_sa3) .gt.0) then
                prorate_ratio = gflk3ad / (dmnd_lec_sa3
     $              + rlocal_storage_3a)
                prorate_ratio = amin1(prorate_ratio,1.0)
              else
                prorate_ratio = 0.0
              endif
              rlok_flow_to_enp_s354 = prorate_ratio * frf
            endif
           if (istaopt(is8bsn-6) .eq. 1 .and. iwsbkpwsta .eq. 1) then
```

```
C
              IF(LWSBPMP.AND.KFLO(IFLS3).EQ.0.) THEN
                 IF (LWSBKP_floor(is8bsn-6) .or. LWSLO_floor(iwca3a))
     $           THEN
                    ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is8bsn-6)
                 ELSE
                    ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is8bsn-6)
                 ENDIF
                 ws_bkp_frac_runff = 1.0
                 IF(.NOT.LWSWCA) THEN
                    KFLO(IFLS3) = - amin1(runoff_south(is8bsn-6)
     $              *ws_bkp_frac *ws_bkp_frac_runff,
     $              rmax_capac_ws_bkp(is8bsn-6))
                    runoff_south(is8bsn-6) = runoff_south(is8bsn-6) -
     $               amin1(runoff_south(is8bsn-6)*ws_bkp_frac
     $               *ws_bkp_frac_runff,rmax_capac_ws_bkp(is8bsn-6))
                 ENDIF
              ENDIF
           endif
C
c
c   route environmental water supply from prop EAA Reservoir
c    to STA34 : Reservoir is primary source.
c
         do irs = 1,nres_ws
           do js = 1,noutlets(irs)
             is = istrctr_ws(ires_no(irs),js)
             if (idshg_opt(ires_no(irs),js) .eq. 3) then
              IF (resname(iwmads(ires_no(irs),is)).eq.
     $        name_res_spec_code(1)) THEN
c             if (res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(19) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(20) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(21) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(22) .or.
c    $         res_out_strname(ires_no(irs),is) .eq.
c    $         res_struc_name_spec_code(23)) then
               if (ifacteaa .eq. 1) then
                 flow_thru_prescap = amax1(grf3amx_prescap - kfloref
     $                 (is8bsn-6) ,0.0)
                 flow_thru_prescap = amin1(flow_thru_prescap,gflk3ad)
                 total_inflow_prescap = runoff_south(is8bsn-6)
     $              + flow_thru_prescap
               else
                 total_inflow_prescap = runoff_south(is8bsn-6)
     $              + gflk3ad
               endif
               if (total_inflow_prescap
     $             .gt. dmnd_lec_sa3) then
                 runoff_to_sta = runoff_south(is8bsn-6)
               else
                 runoff_to_sta = 0.0
               endif
              ELSEIF (resname(iwmads(ires_no(irs),is)).eq.
```

```
     &         name_res_spec_code(4)) THEN
                 runoff_to_sta =inflow_to_sta6_frm_unit1
               ELSE
                 runoff_to_sta = 0.0
               ENDIF
               call ws_from_res(irs,js,ires_no(irs),ieaa_bsn(irs,js)
     $         ,crmin,sumrain,jday,isem_flg,dmnd_sugh,dmnd_sugh_orig
     $         ,capac_semws,capac_sugh,stgref_asr_rec
     $         ,res_outflow_to_basin,stagelo_beg,pctmet
     $         ,sem_dmnd_orig,0,outflow_from_sta2_for_ws
     $         ,wsndes,wsndese,runoff_to_sta,volwtr
     $         ,frac_thru_s8_sta34,wsnde1,wsnde2,wsnde3,res_outflow_env
     $         ,rloc_excess_for_ws,tot_flw_to_c9r_frm_nlkblt
     $         ,ws_from_acmres_lwdd,issm_cutb_opt_bcyp
     $         ,pot_del_to_eaa_basin,alocnnr_potdel
     $         ,total_lec_dmnd_lok,iroute_to_wca3a_frm_holy
     $         ,iopt_env_eaar_thru_g372)
             endif
           enddo
         enddo
c
c    route runoff + flow-through (flow-through may exceed lec demands)
c    to stas if runoff exceeds lec demands
c
            if (istaopt(is8bsn-6) .eq. 1) then
              runoff_s8bsn = runoff_south(is8bsn-6)
              call flow_to_stas (runoff_s8bsn,dmnd_lec_sa3
     $    ,GFLK3AD,1,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $    ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $    ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $    ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $    ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $    ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $    ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $    ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $    ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
            else
              flow_from_eaa_lok_s8 = kflo(ifls8)
     $                 - outflow_from_roten_thru_s8
            endif
c
C DETERMINE THE AMOUNT OF RUNOFF IN MIAMI CANAL BASIN IN THE EAA TO BE
C BACKPUMPED TO LAKE OKEECHOBEE FOR WATER SUPPLY.  WILL BE DONE ONLY
C IF LAKE OKEECHOBEE LEVEL IS LOW AND THERE IS NO W.S. OR ENVIRONMENTAL
C NEEDS IN THE WCA'S OR LOWER EAST COAST
C           IF(LWSBPMP.AND.KFLO(IFLS3).EQ.0.AND.KFLO(IFLS8).GT.0) THEN
            IF (iwsbkpwsta .eq. 0 .or. istaopt(is8bsn-6) .eq. 0) THEN
             IF(LWSBPMP.AND.KFLO(IFLS3).EQ.0) THEN
C WS BP IS DESIRED AND POSSIBLE SINCE S3 IS NOT YET UTILIZED AT THIS POINT
               IF (LWSBKP_floor(is8bsn-6) .or. LWSLO_floor(iwca3a)) THEN
                 ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is8bsn-6)
               ELSE
                 ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is8bsn-6)
               ENDIF
               ws_bkp_frac_runff = 1.0
               IF(.NOT.LWSWCA) THEN
C NO WS DELIVERY IS NEEDED FROM LO TO WCA AND LEC, THUS
```

```
C ALL EAA RO WILL BE BACKPUMPED TO LOK
                  KFLO(IFLS3) = KFLO(IFLS3) - amin1(flow_from_eaa_lok_s8
     $                 *ws_bkp_frac*ws_bkp_frac_runff
     $                 ,rmax_capac_ws_bkp(is8bsn-6))
                  KFLO(IFLS8) = kflo(ifls8)
     $                 - amin1(flow_from_eaa_lok_s8
     $                 *ws_bkp_frac*ws_bkp_frac_runff
     $                 ,rmax_capac_ws_bkp(is8bsn-6))
               ENDIF
             ENDIF
            ENDIF
         ENDIF
      ELSE
C
C FLOOD CONTROL BACKPUMPING FROM EAA TO LOK EXISTS, THUS WATER SUPPLY
C DELIVERY FROM LOK TO WCA AND LEC IS NOT POSSIBLE, AND WATER SUPPLY
C BACKPUMPING IS ALSO NOT POSSIBLE
C
         LWSWCA = .FALSE.
         GFLK3AD = 0.0
         if (iconv_use_inj(is8bsn-6) .eq. 0) then
           KFLOREF(is8bsn-6) = runoff_south(is8bsn-6)
         else
           KFLOREF(is8bsn-6) = runoff_south_bef_divers(is8bsn-6)
         endif
         dmnd_miami_basin = 0.0
         flow_through_ws = gflk3ad
         rlok_flow_to_enp_s354 = 0.0
         if (iconvey_c .ne. 0) then
           lec_dmnd_on_lok(iwca3a,iconvey_c) = 0
         endif
C
c
c   route environmental water supply from prop EAA Reservoir
c    to STA34 : Reservoir is primary source.
c
         do irs = 1,nres_ws
           do js = 1,noutlets(irs)
             is = istrctr_ws(ires_no(irs),js)
             if (idshg_opt(ires_no(irs),js) .eq. 3) then
               IF (resname(iwmads(ires_no(irs),is)) .eq.
     $         name_res_spec_code(1)) THEN
                 runoff_to_sta = runoff_south(is8bsn-6)
               ELSEIF (resname(iwmads(ires_no(irs),is)).eq.
     &         name_res_spec_code(4)) THEN
                 runoff_to_sta =inflow_to_sta6_frm_unit1
               ELSE
                 runoff_to_sta = 0.0
               ENDIF
                call ws_from_res(irs,js,ires_no(irs),ieaa_bsn(irs,js)
     $          ,crmin,sumrain,jday,isem_flg,dmnd_sugh,dmnd_sugh_orig
     $          ,capac_semws,capac_sugh,stgref_asr_rec
     $          ,res_outflow_to_basin,stagelo_beg,pctmet
     $          ,sem_dmnd_orig,0,outflow_from_sta2_for_ws
     $          ,wsndes,wsndese,runoff_to_sta,volwtr
     $         ,frac_thru_s8_sta34,wsnde1,wsnde2,wsnde3,res_outflow_env
     $         ,rloc_excess_for_ws,tot_flw_to_c9r_frm_nlkblt
```

```
     $         ,ws_from_acmres_lwdd,issm_cutb_opt_bcyp
     $         ,pot_del_to_eaa_basin,alocnnr_potdel
     $         ,total_lec_dmnd_lok,iroute_to_wca3a_frm_holy
     $         ,iopt_env_eaar_thru_g372)
             endif
           enddo
         enddo
c
c route runoff into sta34
c
      if (istaopt(is8bsn-6) .eq. 1 ) then
          runoff_s8bsn = runoff_south(is8bsn-6)
          tmpvar = 0.0
          call flow_to_stas(runoff_s8bsn,0.0,tmpvar
     $    ,1,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $    ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $    ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $    ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $    ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $    ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $    ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $    ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $    ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
      endif
      ENDIF
C
C GRF3AMX = CONVEYANCE CAPACITIY BASED ON RATING CURVES
C GRFLK3A = REMAINING CAPACITY AFTER EAA RUNOFF OR DEMAND HAS BEEN CONSIDERED
C WSDOWN = DOWNSTREAM REQUIREMENT TO BE MET BY LAKE OKEECHOBEE
C GFLK3AD = ACTUAL DELIVERY CONSIDERING CONVEYANCE LIMITATIONS BUT
C           WITHOUT S8 SPILLWAY CAPACITY CONSIDERATION
C
C IF LWSWCA = .FALSE. MEANS THAT WATER SUPPLY DELIVERY WILL NOT BE MADE FROM
C LAKE OKEECHOBEE TO WCA AND LEC BECAUSE EITHER:
C   1)  EAA RUNOFF THAT IS RELEASED TO THE SOUTH IS SUFFICIENT TO MEET WS
C       REQUIREMENTS IN THE WCA AND LEC (IF CRMIN() IS USED TO DETERMINE IF WS
C       DELIVERY IS DESIRED FROM THE LAKE TO THE WCA'S AND LEC), OR
C   2)  8.5 FT TRIGGER IS NOT VIOLATED (IF 3-GAGE AVERAGE IS USED TO DETERMINE
C       IF WS DELIVERY IS DESIRED FROM THE LAKE TO THE WCA'S AND LEC) OR
C   3)  WS DELIVERY TO THE SOUTH IS NOT POSSIBLE DUE TO FLOOD CONTROL
C       BACKPUMPING OF EAA RUNOFF TO LAKE OKEECHOBEE
C
C ASSUMPTIONS:
C   * ANY RELEASE FROM LAKE OKEECHOBEE PASSES THROUGH THE S-8 SPILLWAY
C   * EAA RUNOFF  GREATER  THAN 25 PERCENT OF CANAL DESIGN CAPACITY
C   * PASSES THROUGH S8 PUMP ONLY
C     GFLK3AD = WATER SUPPLY DELIVERY TO WCA-3A
C     S8SPMX = S8 SPILLWAY CAPACITY AVAILABLE FOR WATER SUPPLY
c
       if (wcaenv(idn_wca_indx_env(is8bsn-6)) .eq. 'TRUE ') then
          add_inflow = flow_through_ws
       else
          add_inflow = gflk3ad
       endif
c
      if ((istaopt(is8bsn-6) .eq. 0.or.runoff_south(is8bsn-6)
     $ +add_inflow.le.amax1(dmnd_lec_sa3,0.0)).and.flow_to_wca_prior
```

```
     $ .ne. 'LEC' .and. ((((.not. LWSLO_floor(iwca3a) .or. .not.
     $ LWSLECSTA(is8bsn-6)) .or. floor_grid_cell_opt.ne.'MARSH')
     $ .AND. rmin_stg_wca3a .le. rmnstg(iwca3a,iconvey_canal_no_wca
     $ (iwca3a)))) .or. iopt_ws_lec_to_sta_wca_abv_flr .ne. 1) then
c     if ((istaopt(is8bsn-6) .eq. 0.or.runoff_south(is8bsn-6)
c    $    +add_inflow.le.amax1(dmnd_lec_sa3,0.0)).and.flow_to_wca_prior
c    $    .ne. 'LEC') then
       if (istaopt(is8bsn-6) .eq. 0) then
        if (eaa_conv_opt_s8 .ne. 'PUMP') then
         if (kflo(ifls8) .le. s8_cap) then
          rem_sply_cap = amax1(s8_cap - kflo(ifls8),0.0)
          WSLK3A = AMIN1(GFLK3AD,rem_sply_cap)*frac_del_frm_wca(iwca3a)
          rem_cap_sply_s8 = s8_cap - wslk3a - kflo(ifls8)
          rem_cap_sply_s8 = amax1(rem_cap_sply_s8,0.0)
          s8_pump_flow = 0.0
          s8_grav_flow = kflo(ifls8) + wslk3a
         else
          WSLK3A = GFLK3AD*frac_del_frm_wca(iwca3a)
          rem_cap_sply_s8 = 0.0
          total_flow_s8 = kflo(ifls8) + wslk3a
          cuhw = -1.0
          call eaa_prof_lu_table(total_flow_s8,cuhw
     $      ,pmp_int(is8bsn-6),1,ifacteaa)
          s8_spill_hw = pmp_int(is8bsn-6)
          s8_cap  =  s8max
          s354_tw     = amin1(cuhw,rmaxstg(is8bsn-6))
          if (LFORPUMP(is8bsn-6)) s354_tw = amin1(cuhw
     $    ,rmaxstg(is8bsn-6),stagelo+rmax_lift(is8bsn-6))
          s354cap = -1.0
          IF (.not. LFORPUMP(is8bsn-6)) THEN
             call eaaspill(s354cap,stagelo,s354_tw,5,ifacteaa)
          ELSE
             call eaaspill(s354cap,stagelo,s354_tw,9,ifacteaa)
          ENDIF
          miamic_cap_upper_grav = -1.0
          if (s354_tw.gt.pmp_int(is8bsn-6)) then
             call eaa_prof_lu_table(miamic_cap_upper_grav,
     $       s354_tw,pmp_int(is8bsn-6),1,ifacteaa)
          else
             miamic_cap_upper_grav = 0.0
          endif
          s8_pump_flow = kflo(ifls8) + wslk3a
          s8_grav_flow = 0.0
         endif
        else
         if (igravity_s8_nosta .eq. 1) then
          rem_sply_cap = amax1(s8_cap - kflo(ifls8),0.0)
          WSLK3A = AMIN1(GFLK3AD,rem_sply_cap)*frac_del_frm_wca(iwca3a)
          rem_cap_sply_s8 = s8_cap - wslk3a - kflo(ifls8)
          rem_cap_sply_s8 = amax1(rem_cap_sply_s8,0.0)
          s8_pump_flow = 0.0
          s8_grav_flow = kflo(ifls8) + wslk3a
         else
          s8_cap = s8max
          s8_spill_hw = pmp_int(is8bsn-6)
          WSLK3A = GFLK3AD*frac_del_frm_wca(iwca3a)
          s8_pump_flow = kflo(ifls8) + wslk3a
```

```
          s8_grav_flow = 0.0
         endif
        endif
       else
        wslk3a = amin1(gflk3ad,s8_cap-kflo(ifls8))
        wslk3a = amax1(wslk3a,0.0)*frac_del_frm_wca(iwca3a)
        rem_cap_sply_s8 = S8_cap - wslk3a-kflo(ifls8)
        rem_cap_sply_s8 = amax1(rem_cap_sply_s8,0.0)
       endif
       KFLO(IFLS3) = KFLO(IFLS3) + WSLK3A
       KFLO(IFLS8) = KFLO(IFLS8) + WSLK3A
       kfloref(is8bsn-6) = kfloref(is8bsn-6) + WSLK3A
       if (opt_pipeline_ws(is8bsn-6) .eq. 'PIPEL  ') then
         add_ws_pipeline_3a = amin1(kflo(ifls8),runoff_south(is8bsn-6)
     $         ,dmnd_lec_sa3)
         ws_pipeline_3a = ws_pipeline_3a + WSLK3A + add_ws_pipeline_3a
       endif
       DELLEC = DELLEC + WSLK3A * 1.9835
       WSS354 = WSLK3A + WSS354
       if (nint(GFLK3AD) .gt. 0) then
         rlok_flow_to_enp_s354 = rlok_flow_to_enp_s354*(wslk3a/GFLK3AD)
       endif
       ws_to_lec_s354 = amax1(wslk3a - rlok_flow_to_enp_s354,0.0)
       if (istaopt(is8bsn-6) .eq. 1) then
         if ( eaa_conv_opt_s8 .eq. 'PUMP') then
            if (igravity_s8 .eq. 1) then
              s8_pump_flow = 0.0
              s8_grav_flow = kflo(ifls8)
            else
              s8_grav_flow = 0.0
              s8_pump_flow = kflo(ifls8)
            endif
         else
           s8_pump_flow = amax1(kflo(ifls8) - s8_cap ,0.0)
           s8_grav_flow = amin1(kflo(ifls8),s8_cap)
         endif
       endif
      endif
c
      avvol_upstream = -901.
      cname = 'miamic'
      str_name_new_pmp = 'g372  '
      str_name_new = 's8newg'
      cname_upper  = 'miamc1'
      cname_lower  = 'miamc2'
      if (istaopt(is8bsn-6) .eq. 1) then
        if (eaa_conv_opt_s8 .eq. 'PUMP' .or. runoff_south(is8bsn-6)
     $         .gt. avail_cap_for_sta34_miami) then
          miamic_cap       = miamic_cap_upper_pump
          miamic_cap_lower = miamic_cap_lower_pump
          if (igravity_s8 .eq. 1) then
            s8_pump_flow = 0.0
            s8_grav_flow = kflo(ifls8)
          else
            s8_grav_flow = 0.0
            s8_pump_flow = kflo(ifls8)
          endif
```

```
        else
          miamic_cap       = miamic_cap_upper_grav
          miamic_cap_lower = miamic_cap_lower_grav
          s8_pump_flow = amax1(kflo(ifls8)-s8_cap,0.0)
          s8_grav_flow = amin1(kflo(ifls8),s8_cap)
        endif
      else
        miamic_cap       = miamic_cap_upper_grav
        miamic_cap_lower = miamic_cap_lower_grav
      endif
c
      total_flow_s3 = wss354+dmnd_miami_basin
      if (lp(23)) then
       write(62,235) ifyr+iyear-1,month,iday,flnm(ifls3),stagelo
     $  ,s354_tw,s354cap,avvol_upstream,total_flow_s3
     $  ,dmnd_lec_sa3+wsuply(indxsta(is8bsn-6))
     $  +sem_cyp_dmnd(jday)/1.9835+ws_frm_lok_to_semcyp
      endif
      if (istaopt(is8bsn-6) .eq. 0) s8_new_hw = s8_spill_hw
      if (istaopt(is8bsn-6) .eq. 0) then
        total_flow_upper_reach = wss354 + runoff_south(is8bsn-6)
     $           + dmnd_miami_basin
      else
        total_flow_upper_reach = wss354
     $   +runoff_south(is8bsn-6)+dmnd_miami_basin
     $   +add_conv_red_miami
      endif
      if (lp(23)) then
       write(62,235) ifyr+iyear-1,month,iday,cname_upper,s354_tw
     $  ,s8_new_hw,miamic_cap,avvol_upstream
     $  ,total_flow_upper_reach,dmnd_lec_sa3+wsuply(indxsta(is8bsn-6))
     $  +sem_cyp_dmnd(jday)/1.9835+ws_frm_lok_to_semcyp
      endif
      if ( istaopt(is8bsn-6) .eq. 1 ) then
        wsneeds_sta34 = wsdel_to_sta(indxsta(is8bsn-6))
        flow_s8new_to_sta_t = flow_s8new_to_sta + wsneeds_sta34
        if (lp(23)) then
        write(62,235) ifyr+iyear-1,month,iday,str_name_new_pmp,s8_new_hw
     $ ,chdep(ista34),g372cap,avvol_upstream,flow_s8new_to_sta_t
     $ ,wsneeds_sta34
        total_flow_lower = kflo(ifls8) + ws_frm_lok_to_semcyp
     $     - flow_thru_s8/86400.
        write(62,235) ifyr+iyear-1,month,iday,str_name_new,s8_new_hw
     $ ,s8_new_tw,s8_cap_new,avvol_upstream,total_flow_lower
     $ ,dmnd_lec_sa3 + sem_cyp_dmnd(jday)/1.9835+ws_frm_lok_to_semcyp
        write(62,235) ifyr+iyear-1,month,iday,cname_lower,s8_new_tw
     $  ,s8_spill_hw,amax1(flow_lower_reach_max,miamic_cap_lower)
     $  ,avvol_upstream,total_flow_lower+outflow_from_roten_thru_s8
     $  ,dmnd_lec_sa3+sem_cyp_dmnd(jday)/1.9835+ws_frm_lok_to_semcyp
        endif
      endif
      if (lp(23)) then
      write(62,235) ifyr+iyear-1,month,iday,flnm(ifls8),s8_spill_hw
     $  ,s8_tailw,s8_cap,avvol_upstream,kflo(ifls8)
     $  ,dmnd_lec_sa3
      endif
 235  format(i5,2i3,5x,a6,5x,2f7.2,7f10.0)
```

```
      if (opt_pipeline_ws(is8bsn-6) .eq. 'PIPEL  ') then
         kflo(ifls8) = kflo(ifls8) - ws_pipeline_3a
      endif
C
C TO WCA-2A
C
C
         wslec = 0.0
         iserv = iserv_area(is7bsn-6)
         DO j = 1,nsacnl(iserv)
           WSLEC = WSLEC + DQU(isa_canal_indx(iserv,j))
     $          * 86400.
          do i = 1,nsacnl(iserv_area(is8bsn-6))
           if (isa_canal_indx(iserv,j) .eq. isa_canal_indx
     $       (iserv_area(is8bsn-6),i)) then
             WSLEC = WSLEC + DQU(isa_canal_indx(iserv_area(is8bsn-6),i))
     $          * 86400.
           endif
          enddo
         ENDDO
         DO j = 1,nsacnl(iserv)
          do i = 1,nsacnl(iserv_area(is8bsn-6))
           if (isa_canal_indx(iserv,j) .eq. isa_canal_indx
     $       (iserv_area(is8bsn-6),i)) then
            if (wsdown .gt. 0.0) then
             WSLEC = WSLEC - amin1(gflk3ad,wsdown)/wsdown
     $                 * DQU(isa_canal_indx(iserv,j))
            endif
           endif
          enddo
         enddo
         iconvey_c = iconvey_canal_no(is7bsn-6)
         if (iconvey_canal_no(is7bsn-6) .ne. 0) then
          IL38 = intcnl(idn_wca_indx(is7bsn-6)
     &             ,iconvey_canal_no(is7bsn-6))

          floor = floor_elev(iwca2a,iconvey_canal_no(is7bsn-6))
         else
          IL38 = nch + 1
         endif
c
c  if pipeline is used to route water from EAA/LOK to LEC
c
          if (opt_pipeline_ws(is7bsn-6) .eq. 'PIPEL  ') then
             LWSLO_floor(iwca2a) = .TRUE.
             WSWCA2A = 0.0
             GO TO 301
          endif
c
          if (floor_grid_cell_opt .eq. 'MARSH') then
           do i = 1,no_of_subareas_floor(iwca2a)
            if (avg_sim_wca_stg_floor(iwca2a,i) .lt.
     $           avg_wca_subarea_stg_strt_count(iwca2a,i)) then
              no_days_below_floor(iwca2a,i)
     $           = no_days_below_floor(iwca2a,i) + 1
            else
              no_days_below_floor(iwca2a,i) = 0
```

```
        endif
       enddo
       ilok_del_ws = 0
       do i = 1,no_of_subareas_floor(iwca2a)
         if ((no_days_below_floor(iwca2a,i) .gt.
$          min_consec_days_bel_strtstg .and.
$          avg_sim_wca_stg_floor(iwca2a,i) .lt.
$          avg_wca_subarea_crit_stage(iwca2a,i))
$          .or. (avg_sim_wca_stg_floor(iwca2a,i)
$          .le. avg_wca_subarea_stg_disc_lok(iwca2a,i)
$          .and. no_days_deliv_from_lok(iwca2a,i)
$          .gt. 0) .or. LWSLO_floor_temp(iwca2a,i)) then
             LWSLO_floor_temp(iwca2a,i) = .TRUE.
             LWSLO_floor_temp(iwca2a,i) =
$               .NOT. (avg_sim_wca_stg_floor(iwca2a,i)
$               .gt. avg_wca_subarea_stg_disc_lok(iwca2a,i))
         endif
         if (LWSLO_floor_temp(iwca2a,i)) then
           no_days_deliv_from_lok(iwca2a,i) =
$             no_days_deliv_from_lok(iwca2a,i) + 1
           index_lok_deliv(iwca2a,i) = 1
           ilok_del_ws = ilok_del_ws + 1
         else
           no_days_deliv_from_lok(iwca2a,i) = 0
           index_lok_deliv(iwca2a,i) = 0
         endif
       enddo
       LWSLO_floor(iwca2a) = ilok_del_ws .gt. 0
      endif
     if (IL38 .le. nch) then
      WSWCA2A = (floor - CHDEP(IL38))*CL(IL38)*WIDTH(IL38)
&      - CGSEEP(IL38,floor,0,0.) - TOVFLC(IL38,floor,0,0.,0,1)
      if (LWSLO_floor(iwca2a)) then
        RLKDLWS = WSLEC
        if (nint(RLKDLWS/86400.) .gt. 0) then
          index_ws_frm_outside_source(iwca2a,iconvey_c) = 1
        else
          index_ws_frm_outside_source(iwca2a,iconvey_c) = 0
        endif
        rlok_ws_ca2a = WSLEC + AMIN1(WSWCA2A,0.0)
        rlok_ws_ca2a = amax1(rlok_ws_ca2a,0.0)
        if (nint(rlok_ws_ca2a) .gt. 0) then
          index_ws_frm_outside_src_canal(iwca2a,iconvey_c) = 1
        else
          index_ws_frm_outside_src_canal(iwca2a,iconvey_c) = 0
        endif
        logic_floor(iwca2a) = 1
      else
        RLKDLWS = WSLEC + AMIN1(WSWCA2A,0.0)
        RLKDLWS = AMAX1(RLKDLWS,0.0)
        if (nint(RLKDLWS/86400.) .gt. 0) then
          index_ws_frm_outside_src_canal(iwca2a,iconvey_c) = 1
        else
          index_ws_frm_outside_src_canal(iwca2a,iconvey_c) = 0
        endif
        index_ws_frm_outside_source(iwca2a,iconvey_c) = 0
        logic_floor(iwca2a) = 0
```

```
          endif
         flow_thru_s7 = dischg_to_s7_frm_sta34 * 86400.
         rlkdlws = rlkdlws - amin1(flow_thru_s7,rlkdlws)
         RLKDLWS = RLKDLWS - amin1(ws_frm_res_to_wca2a
     $          *86400.,RLKDLWS)
         else
          rlkdlws = 0.0
          if (iconvey_c .ne. 0) then
            index_ws_frm_outside_src_canal(iwca2a,iconvey_c) = 0
            index_ws_frm_outside_source(iwca2a,iconvey_c) = 0
          endif
          if (LWSLO_floor(iwca2a)) then
            logic_floor(iwca2a) = 1
          else
            logic_floor(iwca2a) = 0
          endif
         endif
         go to 302
c
c  for pipeline
c
  301    rlkdlws = wslec
         flow_thru_s7 = dischg_to_s7_frm_sta34 * 86400.
         ws_pipeline_2a = ws_pipeline_2a + amin1(flow_thru_s7,rlkdlws)
     $                      /86400.
         rlkdlws = rlkdlws - amin1(flow_thru_s7,rlkdlws)
         ws_pipeline_2a = ws_pipeline_2a +amin1(ws_frm_res_to_wca2a
     $                      ,rlkdlws/86400.)
         RLKDLWS = RLKDLWS - amin1(ws_frm_res_to_wca2a
     $          *86400.,RLKDLWS)
 302     if (flow_to_wca_prior .eq. 'LEC' .and. istaopt(is7bsn-6)
     $       .eq. 1) then
           if ((iconvey_canal_no(is150bws-6) .eq.
     $         iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $         (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
     $         .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
     $           then
             rem_ws_needs_3a = amax1(dmnd_lec_sa3 - kflo(ifls8)
     $                - ws_pipeline_3a,0.0)
           else
             rem_ws_needs_3a = dmnd_lec_sa3_s150
           endif
           dmnd_lec_sa2 = rlkdlws / 86400. + rem_ws_needs_3a
         else
           dmnd_lec_sa2 = rlkdlws / 86400. + dmnd_lec_sa3_s150
           dmnd_lec_sa2_s7 = rlkdlws / 86400.
           rem_ws_needs_3a = 0.0
         endif
         rlkdlws = dmnd_lec_sa2 * 86400.
         if (ifacteaa .eq. 1) then
           grf2amx_prescap = grf2amx/amax1(rcpfacteaa_canal(is7bsn-6)
     $                      ,1.0)
         endif
         if (is7bsn-6 .ge. 2) then
           do isindx = is7bsn-6,2,-1
            if (ieaa_res_asr(is7bsn-6) .eq. ieaa_res_asr(isindx-1)) then
              inflow_indx = inflow_indx + 1
```

```
            endif
           enddo
         endif
         runoff_south_bef_divers(is7bsn-6) = runoff_south(is7bsn-6)
         runoff_south_bef_divers(is6bsn-6) = runoff_south(is6bsn-6)
     $                                          + divers
c
c  determine if criteria are met for environmental ws releases
c   from LOK to WCA-3A and WCA-2A
c
        ntarg = 0
        if (lrunoff_to_res_when_above_targ .eq. 'FALSE') then
          do itarg = 1,no_targ_loc(is7bsn-6)
            if (LWSLO(itarg_area(itarg,is7bsn-6))) then
               ntarg = ntarg + 1
            endif
          enddo
        else
           do itarg = 1,no_targ_loc(is7bsn-6)
             if (avg_sim_wca_stg(itarg_area(itarg,is7bsn-6))
     $         .lt. stage_import_wca(itarg_area(itarg,is7bsn-6))
     $         + offset(itarg_area(itarg,is7bsn-6),1)) then
                ntarg = ntarg + 1
             endif
           enddo
        endif
c
        ntarg_hill = 0
        do itarg = 1,no_targ_loc(is6bsn-6)
          if (LWSLO(itarg_area(itarg,is6bsn-6))) then
             ntarg_hill = ntarg_hill + 1
          endif
        enddo
c
        if (wcaenv(idn_wca_indx_env(is7bsn-6)) .eq. 'FALSE'.or.
     $         nnrctwca3a .eq. 'FALSE') then
c
c route excess eaa runoff to asr reservoir
c
         if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa(is7bsn-6)
     $         .eq. 1) then
              total_demand_dwn = dmnd_lec_sa2 + amax1(dmnd_lec_sa3
     $             - frf - kflo(ifls8) - ws_pipeline_3a,0.0)
            if ((iconvey_canal_no(is150bws-6) .ne.
     &        iconvey_canal_no(is8bsn-6) .and.
     &        idn_wca_indx(is150bws-6) .eq. idn_wca_indx(is8bsn-6)
     &         .and. iconvey_canal_no(is150bws-6) .ne. 0)
     &         .or. opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
               total_demand_dwn = dmnd_lec_sa2
            endif
            runoff_to_inject = amax1(runoff_south(is7bsn-6)
     $         -total_demand_dwn-wsuply(indxsta(is7bsn-6)),0.0)
            flow_to_eaa_res_rnff(is7bsn-6) = amin1(runoff_to_inject
     $       *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $       ,weircff(ieaa_res_asr(is7bsn-6),inflow_indx,1))
            rem_weir_capac(is7bsn-6)=weircff(ieaa_res_asr(is7bsn-6)
     $          ,inflow_indx,1) - flow_to_eaa_res_rnff(is7bsn-6)
```

```
            volwtr(ieaa_res_asr(is7bsn-6))
     $         = volwtr(ieaa_res_asr(is7bsn-6)) - flow_to_eaa_res_rnff
     $         (is7bsn-6)
            runoff_south(is7bsn-6) = runoff_south(is7bsn-6)
     $          - flow_to_eaa_res_rnff(is7bsn-6)
            flow_to_eaa_res_nnr = flow_to_eaa_res_rnff(is7bsn-6)
           if (istaopt(is7bsn-6) .eq. 0) then
            if (runoff_s7+runoff_s150 .gt. 0.0) then
             ratio_rf = runoff_s7/(runoff_s7+runoff_s150)
            else
             ratio_rf = 0.0
            endif
            q = runoff_s7 - ratio_rf * flow_to_eaa_res_rnff(is7bsn-6)
     $             + dischg_to_s7_frm_sta34
            if (q .gt. s7max) then
              KFLO(IFLS7) = S7MAX
              IAGFLO(IAG(IS7BSN),1) = NINT(Q - S7MAX) + IAGFLO(IAG
     $               (IS7BSN),1)
            ELSE
              KFLO(IFLS7) = Q
            ENDIF
            qs150  = runoff_s150-(1.0-ratio_rf)*flow_to_eaa_res_rnff
     $                  (is7bsn-6)
            kflo(ifls150) = qs150
           endif
          endif
         else
c
c route excess eaa runoff to asr reservoir
c
C
          if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_eaa
     $         (is7bsn-6) .eq. 1) then
            total_demand_dwn = dmnd_lec_sa2 + amax1(dmnd_lec_sa3
     $         - frf - kflo(ifls8) - ws_pipeline_3a,0.0)
            if ((iconvey_canal_no(is150bws-6) .ne.
     &        iconvey_canal_no(is8bsn-6) .and.
     &        idn_wca_indx(is150bws-6) .eq. idn_wca_indx(is8bsn-6)
     &         .and. iconvey_canal_no(is150bws-6) .ne. 0)
     &         .or. opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
             total_demand_dwn = dmnd_lec_sa2
            endif
            if ( .not.(ntarg .gt. 0.and.nnrctwca3a.eq.'TRUE ')) then
             runoff_to_inject = amax1(runoff_south(is7bsn-6)
     $          - total_demand_dwn-wsuply(indxsta(is7bsn-6)),0.0)
            else
             rmax_flow_thru = amin1(s351mx - amax1(kflo(ifls2),0.0)
     $          -wsuply(indxsta(is7bsn-6)),grf2amx
     $          -amax1(dmnd_nnr,0.0)-wsuply(indxsta(is7bsn-6))
     $          ,s7max)
             rmax_flow_thru = amax1(rmax_flow_thru,0.0)
             rmax_flow_bypass_res =amax1(rmax_flow_thru,total_demand_dwn)
             runoff_to_inject = amax1(runoff_south(is7bsn-6)
     $                     - rmax_flow_bypass_res,0.0)
             if( divers_excess_to_res .eq. 'FALSE') then
               runoff_to_inject = 0.0
             endif
```

```
          endif
          flow_to_eaa_res_rnff(is7bsn-6) = amin1(runoff_to_inject
     $       *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $       ,weircff(ieaa_res_asr(is7bsn-6),inflow_indx,1))
          rem_weir_capac(is7bsn-6) =weircff(ieaa_res_asr(is7bsn-6)
     $          ,inflow_indx,1) - flow_to_eaa_res_rnff(is7bsn-6)
          volwtr(ieaa_res_asr(is7bsn-6))
     $         = volwtr(ieaa_res_asr(is7bsn-6)) - flow_to_eaa_res_rnff
     $           (is7bsn-6)
          flow_to_eaa_res_nnr = flow_to_eaa_res_rnff(is7bsn-6)
          runoff_south(is7bsn-6) = runoff_south(is7bsn-6)
     $          - flow_to_eaa_res_rnff(is7bsn-6)
         if (istaopt(is7bsn-6) .eq. 0) then
          if (runoff_s7+runoff_s150 .gt. 0.0) then
           ratio_rf = runoff_s7/(runoff_s7+runoff_s150)
          else
           ratio_rf = 0.0
          endif
          q = runoff_s7 - ratio_rf * flow_to_eaa_res_rnff(is7bsn-6)
     $           + dischg_to_s7_frm_sta34
          if (q .gt. s7max) then
            KFLO(IFLS7) = S7MAX
            IAGFLO(IAG(IS7BSN),1) = NINT(Q - S7MAX) + IAGFLO(IAG
     $              (IS7BSN),1)
          ELSE
            KFLO(IFLS7) = Q
          ENDIF
          qs150  = runoff_s150-(1.0-ratio_rf)*flow_to_eaa_res_rnff
     $                (is7bsn-6)
          kflo(ifls150) = qs150
         endif
        endif
       endif
       runoff_thru_s150 = kflo(ifls150)
c
       ntarg = 0
       do itarg = 1,no_targ_loc(is7bsn-6)
         if (LWSLO(itarg_area(itarg,is7bsn-6))) then
            ntarg = ntarg + 1
         endif
       enddo
c
      KBFBPS2 = KFLO(IFLS2)
      conv_asr = 0.
      iter_asr = 1
      rem_cap_s351_diff_prev = 1.
      flow_incr_init = 400.
      flow_incr  = flow_incr_init
 777  IF(KFLO(IFLS2).GE.0.) THEN
        if (iter_asr .gt. 1 .or. conv_asr .eq. 1.) go to 778
        if (iconv_use_inj(is7bsn-6) .eq. 0) then
          kfloref(is7bsn-6) = dmnd_nnr+runoff_south(is7bsn-6)
        else
          kfloref(is7bsn-6) = dmnd_nnr+runoff_south_bef_divers
     $              (is7bsn-6)
        endif
        iconv_red_nnrc = 0
```

```
          do irs = 1,nres_ws
           if (iconv_use_opt_ws(irs) .eq. 1) then
             do js = 1,noutlets(irs)
               if (ieaa_bsn(irs,js) .eq. is7bsn-6) then
                 iconv_red_nnrc = iconv_red_nnrc + 1
               endif
             enddo
           endif
          enddo
          if (iconv_red_nnrc .gt. 0) kfloref(is7bsn-6) =
     $             kfloref(is7bsn-6)+ dmnd_nnr_orig
     $             - dmnd_nnr_aft_resasr
          IF(GRF2AMX.GT.0.0) THEN
             GRFLK2A = AMAX1(GRF2AMX - kfloref(is7bsn-6),0.0)
          ELSE
             GRFLK2A = 0.0
          ENDIF
          if (istaopt(is7bsn-6) .eq. 0) then
           if (eaa_conv_opt_s7 .eq. 'PUMP') then
            if (((dmnd_lec_sa2_s7 .gt. grf2amx .or.runoff_south(is7bsn-6)
     $           .gt. grf2amx) .and. grf2amx .lt. grf2amx_pump
     $           .and. s351mx .lt. s351cap_pump) .or. dmnd_lec_sa2_s7
     $           .gt. s7_cap) THEN
               grflk2a = amax1(grf2amx_pump - kfloref(is7bsn-6),0.0)
               igravity_s7_nosta = 0
               if (s351mx .lt. s351cap_pump) then
                 s351mx  = s351cap_pump
               endif
               s351_tw = s351_tw_pump
               nnrc_cap_upper_grav = grf2amx_pump
               fllks6 = fllks6_pump
            else
             igravity_s7_nosta = 1
            endif
           else
             igravity_s7_nosta = 1
           endif
          endif
          if (nint(alocnnr/86400.) .gt. 0) then
            dmnd_hill = kbfbps2 - dmnd_nnr
          else
            if (nint(kbfbps2) .gt. 0) then
              dmnd_hill = kbfbps2 - alocnnr/86400.
            else
              dmnd_hill =  0.0
            endif
          endif
          dmnd_hill = amax1(dmnd_hill,0.0)
c
c  deliver water supply from Lake Okeechobee into STA34 via
c  NNR Canal
c
          if (istaopt(is7bsn-6) .eq. 1) then
           ireach = 6
           ctw = pmp_int_new(is7bsn-6)
           if (.not. LFORPUMP(is7bsn-6)) then
             istype = 1
```

```
          istr_eaa = 6
          chw = stagelo
          delta0 = (chw - ctw) / 25.0
          call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo
     $          ,delta0,1.0,100,nnrc_cap_wstosta,str_cap,str_tw
     $          ,ifacteaa)
          if (str_tw .gt. rmaxstg(is7bsn-6)) then
            str_tw = rmaxstg(is7bsn-6)
            str_cap = -1.0
            call eaaspill(str_cap,stagelo,rmaxstg(is7bsn-6),6
     $      ,ifacteaa)
            qcanal = -1.0
            if (str_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,str_tw,ctw,
     &          ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            nnrc_cap_wstosta = qcanal
          endif
        else
          istr_eaa = 10
          str_tw=amin1(stagelo+rmax_lift(is7bsn-6),rmaxstg(is7bsn-6))
          str_cap = -1.0
          call eaaspill(str_cap,stagelo,str_tw,istr_eaa,ifacteaa)
          qcanal = str_cap
          str_tw_max = str_tw
          str_tw = -1.0
          call eaa_prof_lu_table(qcanal,str_tw,ctw,ireach,ifacteaa)
          if (str_tw .gt. str_tw_max) then
            str_tw = str_tw_max
            qcanal = -1.0
            call eaa_prof_lu_table(qcanal,str_tw,ctw,ireach,ifacteaa)
          endif
          nnrc_cap_wstosta = qcanal
        endif
        rem_cap_wstosta = nnrc_cap_wstosta - kfloref(is7bsn-6)
        rem_cap_wstosta = amax1(rem_cap_wstosta,0.0)
        rem_cap_struc = amax1(str_cap - amax1(kflo(ifls2),0.0),0.0)
        wsdel_to_sta(indxsta(is7bsn-6)) = amin1(wsuply(indxsta
     $              (is7bsn-6)) ,rem_cap_wstosta,rem_cap_struc)
        kflo(ifls2) = kflo(ifls2)
     $          + wsdel_to_sta(indxsta(is7bsn-6))
        grflk2a = grflk2a - amin1(wsdel_to_sta(indxsta(is7bsn-6))
     $                        ,grflk2a)
        wss351 = wss351 + wsdel_to_sta(indxsta(is7bsn-6))
        ws_to_sta34_via_nnrc = wsdel_to_sta(indxsta(is7bsn-6))
        kfloref(is7bsn-6) = kfloref(is7bsn-6)
     $         + wsdel_to_sta(indxsta(is7bsn-6))
        wsuply(indxsta(is7bsn-6)) = wsuply(indxsta(is7bsn-6))
     $        - wsdel_to_sta(indxsta(is7bsn-6))
        ws_to_sta34_frm_lok = ws_to_sta34_frm_lok
     $        + wsdel_to_sta(indxsta(is7bsn-6))
      endif
C
      if (indxsta(is7bsn-6) .eq.indxsta(is8bsn-6))
     $     wsdel_to_sta(indxsta(is7bsn-6)) = ws_to_sta34_frm_lok
```

```
C
         IF (.NOT. opt_l8_prop) THEN
           WSNEED_TOL8 = WSNEEDL8
         ELSE
           WSNEED_TOL8 = DQU(IL8S)
         ENDIF
         wslec_sa1 = 0.0
         if (opt_for_hill_bypass .eq. 'TRUE ') then
           iserv = iserv_area(is6bsn-6)
           DO j = 1,nsacnl(iserv)
             WSLEC_SA1 = WSLEC_SA1 + DQU(isa_canal_indx(iserv,j))
     $          * 86400.
           ENDDO
           WSLEC_SA1 = WSLEC_SA1 + (WSNDE1 + WSNDE2 + WSNDE3
     $       +  WSNDES + WSNDESE + WSNEED_TOL8) * 86400.
         else
           WSLEC_SA1 = (WSNDE1 + WSNDE2 + WSNDE3 + WSNDES
     $             + WSNDESE + WSNEED_TOL8) * 86400.
     $             + WSLEC_SA1
         endif
C
         if (istaopt(is5absn-6) .eq. 1 .or. istaopt(5) .eq. 1) then
           rinflow_to_wca1 = 0.0
           if (istaopt(is5absn-6) .eq. 1) then
            rinflow_to_wca1 = out_to_wca1_frm_sta1w
            if (istaopt(5) .eq. 0) then
               untreated_runoff_thru_s5a =addflw(idn_wca_indx(is5absn-6)
     $          ,iconvey_canal_no(is5absn-6))
            endif
           endif
           if (istaopt(5) .eq. 1) then
            rinflow_to_wca1 = out_to_wca1_frm_sta1e + rinflow_to_wca1
            untreated_runoff_thru_s5a = 0.0
           endif
         else
           rinflow_to_wca1 = 0.0
           untreated_runoff_thru_s5a = amax1(kflo(ifls5a5),0.0)
         endif
c
         if (opt_pipeline_ws(is6bsn-6) .eq. 'PIPEL  ') then
            LWSLO_floor(iwca1) = .TRUE.
            WSWCA1 = 0.0
c
c  if water supply deliveries from WCA1 do not occur daily,then
c  LOK deliveries must be limited to capacity of WCA1 to deliver
c  water when demands in SA1 are high
c
             IF (LFRACBFLR(iwca1)) THEN
                no_of_days_del_week = no_of_days_wk_wcadel(iwca1,2)
             ELSE
                no_of_days_del_week = no_of_days_wk_wcadel(iwca1,1)
             ENDIF
             IF (no_of_days_del_week .LE. 5) THEN
                sum_inflow_lwdd_pump_cap = CAPACCS2 + CAPACCS9
     &          + CAPACCS12 + CAPACCS17W + CAPACCS17E + g94abcapac
     &          -(kflo(iflcs12) + kflo(iflcs17w) + kflo(iflcs17e)
     &          + kflo(iflcs9) + kflo(iflcs2))
```

```
              rlok_pot_del = sum_inflow_lwdd_pump_cap
     &        + WSNEED_TOL8
              IF (opt_for_hill_bypass .eq. 'TRUE ') then
                 iserv = iserv_area(is6bsn-6)
                 DO j = 1,nsacnl(iserv)
                   rlok_pot_del = rlok_pot_del +
     $              DQU(isa_canal_indx(iserv,j))
                 ENDDO
              ENDIF
              rlok_pot_del = amin1(rlok_pot_del,WSLEC_SA1/86400.)
           ELSE
             rlok_pot_del = WSLEC_SA1/86400.
           ENDIF
          GO TO 401
        endif
c
         if (floor_grid_cell_opt .eq. 'MARSH') then
          do i = 1,no_of_subareas_floor(iwca1)
           if (avg_sim_wca_stg_floor(iwca1,i) .lt.
     $           avg_wca_subarea_stg_strt_count(iwca1,i)) then
             no_days_below_floor(iwca1,i)
     $          = no_days_below_floor(iwca1,i) + 1
           else
             no_days_below_floor(iwca1,i) = 0
           endif
          enddo
          ilok_del_ws = 0
          do i = 1,no_of_subareas_floor(iwca1)
            if ((no_days_below_floor(iwca1,i) .gt.
     $        min_consec_days_bel_strtstg .and.
     $        avg_sim_wca_stg_floor(iwca1,i) .lt.
     $        avg_wca_subarea_crit_stage(iwca1,i))
     $        .or. (avg_sim_wca_stg_floor(iwca1,i)
     $        .le. avg_wca_subarea_stg_disc_lok(iwca1,i)
     $        .and. no_days_deliv_from_lok(iwca1,i)
     $        .gt. 0) .or. LWSLO_floor_temp(iwca1,i)) then
              LWSLO_floor_temp(iwca1,i) = .TRUE.
              LWSLO_floor_temp(iwca1,i) =
     $           .NOT. (avg_sim_wca_stg_floor(iwca1,i)
     $           .gt. avg_wca_subarea_stg_disc_lok(iwca1,i))
            endif
            if (LWSLO_floor_temp(iwca1,i)) then
              no_days_deliv_from_lok(iwca1,i) =
     $          no_days_deliv_from_lok(iwca1,i) + 1
              index_lok_deliv(iwca1,i) = 1
              ilok_del_ws = ilok_del_ws + 1
            else
              no_days_deliv_from_lok(iwca1,i) = 0
              index_lok_deliv(iwca1,i) = 0
            endif
          enddo
          LWSLO_floor(iwca1) = ilok_del_ws .gt. 0
         endif
c
        if (iconvey_canal_no(is6bsn-6) .ne. 0) then
          IWCA = idn_wca_indx(is6bsn-6)
          ICA1 = intcnl(idn_wca_indx(is6bsn-6)
```

```
     &              ,iconvey_canal_no(is6bsn-6))
            floor = floor_elev(iwca,iconvey_canal_no(is6bsn-6))
            iconvey_c = iconvey_canal_no(is6bsn-6)
            WSWCA1 = (floor-CHDEP(ICA1))*CL(ICA1)*WIDTH(ICA1) -
     &       CGSEEP(ICA1,floor,0,0.)-TOVFLC(ICA1,floor,0,0.,0,1)
     &       + rlvseep_frm_cnl(ica1)*86400.
            WSWCA1_floorelev = (rmnstg(iwca,iconvey_c) - CHDEP(ICA1))
     &       *CL(ICA1)*WIDTH(ICA1) - CGSEEP(ICA1,rmnstg(iwca,iconvey_c)
     &       ,0,0.) - TOVFLC(ICA1,rmnstg(iwca,iconvey_c),0,0.,0,1)
     &       + rlvseep_frm_cnl(ica1)*86400.
c
c  if water supply deliveries from WCA1 do not occur daily,then
c  LOK deliveries must be limited to capacity of WCA1 to deliver
c  water when demands in SA1 are high
c
          IF (LFRACBFLR(iwca1)) THEN
             no_of_days_del_week = no_of_days_wk_wcadel(iwca1,2)
          ELSE
             no_of_days_del_week = no_of_days_wk_wcadel(iwca1,1)
          ENDIF
          IF (no_of_days_del_week .LE. 5) THEN
             sum_inflow_lwdd_pump_cap = CAPACCS2 + CAPACCS9
     &        + CAPACCS12 + CAPACCS17W + CAPACCS17E + g94abcapac
     &        -(kflo(iflcs12) + kflo(iflcs17w) + kflo(iflcs17e)
     &        + kflo(iflcs9) + kflo(iflcs2))
             rlok_pot_del = sum_inflow_lwdd_pump_cap
     &       + WSNEED_TOL8
             IF (opt_for_hill_bypass .eq. 'TRUE ') then
                iserv = iserv_area(is6bsn-6)
                DO j = 1,nsacnl(iserv)
                  rlok_pot_del = rlok_pot_del +
     $             DQU(isa_canal_indx(iserv,j))
                ENDDO
             ENDIF
             rlok_pot_del = amin1(rlok_pot_del,WSLEC_SA1/86400.)
          ELSE
            rlok_pot_del = WSLEC_SA1/86400.
          ENDIF
          rlkdlws_floor_elev = rlok_pot_del*86400.
     &    + amin1(WSWCA1_floorelev,0.0)
          rlkdlws_floor_elev = amax1(rlkdlws_floor_elev,0.0)
          if (LWSLO_floor(iwca1)) then
           RLKDLWS_SA1 = rlok_pot_del*86400.
           if (nint(RLKDLWS_SA1/86400.) .gt. 0) then
             index_ws_frm_outside_source(iwca1
     &              ,iconvey_canal_no(is6bsn-6)) = 1
             rlok_ws_ca1 = rlok_pot_del*86400. + AMIN1(WSWCA1,0.0)
             rlok_ws_ca1 = amax1(rlok_ws_ca1,0.0)
             if (lok_criteria_wca1 .eq. 1) then
                if (nint(rlok_ws_ca1/86400.) .gt. 0) then
                  lok_criteria_wca1_lok_ws = 1
                  if (nint(rlkdlws_floor_elev/86400.) .gt. 0) then
                    index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 1
                  else
                    index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 0
```

```
                  endif
                else
                  lok_criteria_wca1_lok_ws = 0
                  index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 0
                endif
              else
                lok_criteria_wca1_lok_ws = 0
                if (nint(rlok_ws_ca1/86400.) .gt. 0) then
                  index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 1
                else
                  index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 0
                endif
              endif
            else
              lok_criteria_wca1_lok_ws = 0
            endif
            logic_floor(iwca1) = 1
          else
C RLKDLWS: TOTAL DEMAND TO BE MET BY LAKE OKEECHOBEE TO WCA-1
            RLKDLWS_SA1 = rlok_pot_del*86400. + AMIN1(WSWCA1,0.0)
            RLKDLWS_SA1 = AMAX1(RLKDLWS_SA1,0.0)
            if (nint(RLKDLWS_SA1/86400.) .gt. 0) then
              if (lok_criteria_wca1 .eq. 1) then
                if (nint(rlkdlws_floor_elev/86400.) .gt. 0) then
                  index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 1
                else
                  index_ws_frm_outside_src_canal(iwca1
     &                   ,iconvey_canal_no(is6bsn-6)) = 0
                endif
                lok_criteria_wca1_lok_ws = 1
              else
                lok_criteria_wca1_lok_ws = 0
                index_ws_frm_outside_src_canal(iwca1
     &                  ,iconvey_canal_no(is6bsn-6)) = 1
              endif
            else
              index_ws_frm_outside_src_canal(iwca1
     &               ,iconvey_canal_no(is6bsn-6)) = 0
              lok_criteria_wca1_lok_ws = 0
            endif
            index_ws_frm_outside_source(iwca1
     &               ,iconvey_canal_no(is6bsn-6)) = 0
            logic_floor(iwca1) = 0
          endif
          RLKDLWS_SA1 = RLKDLWS_SA1 - amin1(rinflow_to_wca1*86400.
     $           ,RLKDLWS_SA1)
          RLKDLWS_SA1 = RLKDLWS_SA1 - amin1(ws_frm_res_to_wca1
     $           *86400.,RLKDLWS_SA1)
          RLKDLWS_SA1_WO_UNTRTRFS5A = RLKDLWS_SA1/86400.
          RLKDLWS_SA1 = RLKDLWS_SA1 - AMIN1(RLKDLWS_SA1,
     $           untreated_runoff_thru_s5a*86400.)
          else
           ICA1 = nch+1
```

```
          rlkdlws_sa1 = 0.0
          ws_down_s6  = 0.0
          RLKDLWS_SA1_WO_UNTRTRFS5A = 0.0
          lok_criteria_wca1_lok_ws = 0
          if (LWSLO_floor(iwca1)) then
            logic_floor(iwca1) = 1
          else
            logic_floor(iwca1) = 0
          endif
         endif
         go to 402
 401     rlkdlws_sa1 = rlok_pot_del*86400.
         ica1 = nch+1
         ws_pipeline_ca1 = ws_pipeline_ca1 + amin1(RLKDLWS_SA1
     $           /86400.,rinflow_to_wca1)
         RLKDLWS_SA1 = RLKDLWS_SA1 - amin1(rinflow_to_wca1*86400.
     $          ,RLKDLWS_SA1)
         ws_pipeline_ca1 = ws_pipeline_ca1 + amin1(RLKDLWS_SA1/86400.
     $          ,untreated_runoff_thru_s5a)
         RLKDLWS_SA1 = RLKDLWS_SA1 - AMIN1(RLKDLWS_SA1,
     $          untreated_runoff_thru_s5a*86400.)
         ws_pipeline_ca1 = ws_pipeline_ca1 + amin1(RLKDLWS_SA1/86400.
     $          ,ws_frm_res_to_wca1)
         RLKDLWS_SA1 = RLKDLWS_SA1 - amin1(ws_frm_res_to_wca1
     $          *86400.,RLKDLWS_SA1)
 402     rlok_to_sa1 = rlkdlws_sa1 / 86400.
C
         tot_ws_down_s6 = dqu(lhl) + wsndes + wsndese
         if (nint(rlkdlws_sa1/86400.) .gt. 0 .and. opt_for_hill_bypass
     $      .eq. 'TRUE ') then
            ratio_rem =rlkdlws_sa1/wslec_sa1
            ratio_rem = amin1(ratio_rem,1.0)
            ws_down_s6 = tot_ws_down_s6*ratio_rem
         elseif (opt_for_hill_bypass .eq. 'FALSE') then
            rdlws_sta2 = rlok_pot_del/86400. + dqu(lhl)
     $        + dqu(iacmdr) + amin1(wswca1,0.0)/86400.
            rdlws_sta2 = amax1(rdlws_sta2,0.0)
            rdlws_sta2 = rdlws_sta2 - amin1(rinflow_to_wca1,rdlws_sta2)
            if (nint(wslec_sa1/86400.+dqu(lhl)+dqu(iacmdr)) .gt. 0) then
             ratio_rem_sta2 =rdlws_sta2/(wslec_sa1/86400.
     $           +dqu(lhl)+dqu(iacmdr))
             ratio_rem_sta2 = amin1(ratio_rem_sta2,1.0)
            else
             ratio_rem_sta2 = 0.0
            endif
            ws_down_sta2 = tot_ws_down_s6*ratio_rem_sta2
            if (istaopt(is6bsn-6) .eq. 1) then
             do irs = 1,nres_ws
             do js = 1,noutlets(irs)
              if (idshg_opt(ires_no(irs),js) .ne. 1
     $          .and. idshg_opt(ires_no(irs),js) .ne. 3) then
               call ws_from_res(irs,js,ires_no(irs),ieaa_bsn(irs,js)
     $         ,crmin,sumrain,jday,isem_flg,dmnd_sugh,dmnd_sugh_orig
     $         ,capac_semws,capac_sugh,stgref_asr_rec
     $         ,res_outflow_to_basin,stagelo_beg,pctmet
     $         ,sem_dmnd_orig,1,outflow_from_sta2_for_ws
     $         ,wsndes,wsndese,runoff_to_sta,volwtr
```

```
     $         ,frac_thru_s8_sta34,wsnde1,wsnde2,wsnde3,res_outflow_env
     $         ,rloc_excess_for_ws,tot_flw_to_c9r_frm_nlkblt
     $         ,ws_from_acmres_lwdd,issm_cutb_opt_bcyp
     $         ,pot_del_to_eaa_basin,alocnnr_potdel
     $         ,total_lec_dmnd_lok,iroute_to_wca3a_frm_holy
     $         ,iopt_env_eaar_thru_g372)
              endif
             enddo
             enddo
            endif
            ws_down_s6 = 0.0
         endif
         dmnd_lec_s6 = ws_down_s6
         if (ifacteaa .eq. 1) then
           fllks6_prescap = fllks6 / amax1(rcpfacteaa_canal(is6bsn-6)
     $                       ,1.0)
         endif
       runoff_south_s6bsin = runoff_south(is6bsn-6)+divers
       divers_south = divers
c
       tmp_flw_s2 = kflo(ifls2)
 778   if  (wcaenv(idn_wca_indx_env(is6bsn-6)) .eq. 'FALSE') then
c
c route eaa runoff to asr reservoir if there are no LEC demands and
c no environmental demands in everglades
c
        if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_hill
     $         .eq. 1) then
           runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $           - dmnd_lec_s6-wsuply(indxsta(is6bsn-6))
     $           - wsuply(indx_sta2w_frm_lok_ws),0.0)
           res_flow = amin1(runoff_to_inject
     $      *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $      ,rem_weir_capac(is7bsn-6))
           divers_to_res = amin1(divers,res_flow)
           runoff_south_s6bsin = runoff_south(is6bsn-6)
     $         - (res_flow - divers_to_res)
        endif
       else
c
c route eaa runoff to asr reservoir if there are no LEC demands and
c no environmental demands in everglades
c
          if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_hill
     $         .eq. 1) then
           if (ntarg_hill .eq. 0) then
             runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $         - dmnd_lec_s6-wsuply(indxsta(is6bsn-6))
     $         - wsuply(indx_sta2w_frm_lok_ws),0.0)
           else
             rem_cap_s351_bef = s351mx - tmp_flw_s2
     $           - wsuply(indxsta(is6bsn-6))
             rmax_flow_thru = amin1(rem_cap_s351_bef
     $        ,fllks6 - dmnd_hill-wsuply(indxsta(is6bsn-6))
     $        ,s6max)
             rmax_flow_thru = amax1(rmax_flow_thru,0.0)
             rmax_flow_bypass_res = amax1(rmax_flow_thru,dmnd_lec_s6)
```

```
              runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $                - rmax_flow_bypass_res,0.0)
              if (divers_excess_to_res .eq. 'FALSE') then
                runoff_to_inject = 0.0
              endif
            endif
            res_flow = amin1(runoff_to_inject
     $       *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $       ,rem_weir_capac(is7bsn-6))
            divers_to_res = amin1(divers,res_flow)
            divers_south = divers - divers_to_res
            runoff_south_s6bsin = runoff_south(is6bsn-6)
     $          - (res_flow - divers_to_res)
          endif
        endif
        flow_s2_tmp = kflo(ifls2)
c
          add_red_in_conv = 0.0
          do irs = 1,nres_ws
           if (iconv_use_opt_ws(irs) .eq. 1) then
             do js = 1,noutlets(irs)
               if (ieaa_bsn(irs,js) .eq. is7bsn-6) then
                 add_red_in_conv = add_red_in_conv +
     $            res_outflow_to_basin(irs,js)
               endif
             enddo
           endif
          enddo
          add_red_in_conv = add_red_in_conv - dmnd_nnr_orig
     $             + dmnd_nnr_aft_resasr
          add_red_in_conv = amax1(add_red_in_conv,0.0)
          if (iconv_use_inj(is6bsn-6) .eq. 0) then
           rem_canal_cap_hill = fllks6 - amax1(runoff_south_s6bsin
     $     + divers_south,dmnd_hill+add_red_in_conv+divers_south)
          else
           rem_canal_cap_hill = fllks6 - (runoff_south_bef_divers
     $       (is6bsn-6)+dmnd_hill+add_red_in_conv)
          endif
          rem_canal_cap_hill = amax1(rem_canal_cap_hill,0.0)
          total_rem_cap_nnrhill = grflk2a + rem_canal_cap_hill
          if (total_rem_cap_nnrhill .gt. 0.0) then
            ratio_cap_desgn = (s351mx-kflo(ifls2))
     $                        / total_rem_cap_nnrhill
            ratio_cap_desgn = amin1(ratio_cap_desgn,1.0)
            ratio_cap_desgn = amax1(ratio_cap_desgn,0.0)
          else
            ratio_cap_desgn = 1.0
          endif
C
C CONSIDER THE POTENTIAL FOR WATER SUPPLY BACKPUMPING
C
C23456789*123456789*123456789*123456789*123456789*123456789*123456789*12
         if (wcaenv(idn_wca_indx_env(is7bsn-6)) .eq. 'FALSE'.or.
     $            nnrctwca3a .eq. 'FALSE') then
c
c  route runoff from nnrc basin into sta34
c
```

```
      if (istaopt(is7bsn-6) .eq. 1 .and. iwsbkpwsta .eq. 1) then
C
       IF(nint(RLKDLWS/86400.).GE.0) THEN
         RLKDLCFS = RLKDLWS/86400.
         IF(runoff_south(is7bsn-6) .GT. RLKDLCFS) THEN
            IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
              IF (LWSBKP_floor(is7bsn-6) .or. LWSLO_floor(iwca2a))
     $         THEN
               ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
              ELSE
               ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
              ENDIF
              ws_bkp_frac_runff = 1.0
               KFLO(IFLS2) = - amin1((runoff_south(is7bsn-6) -
     $         RLKDLCFS)*ws_bkp_frac *ws_bkp_frac_runff,
     $         rmax_capac_ws_bkp(is7bsn-6))
               runoff_south(is7bsn-6) = runoff_south(is7bsn-6) -
     $         amin1((UNTRTRF_S7 - dmnd_lec_sa2_s7)*ws_bkp_frac
     $         *ws_bkp_frac_runff,rmax_capac_ws_bkp(is7bsn-6))
            ENDIF
         ENDIF
         RLKDLWS = RLKDLCFS*86400.
       ENDIF
      endif
      if (istaopt(is7bsn-6) .eq. 1) then
       runoff_s7bsn = runoff_south(is7bsn-6)
        tmpvar = 0.0
        call flow_to_stas(runoff_s7bsn,RLKDLWS/86400.
     $ ,tmpvar,2,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $ ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $ ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $ ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $ ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $ ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $ ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $ ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $ ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
      endif
      if (iwsbkpwsta .eq. 1 .and. istaopt(is7bsn-6) .eq. 1)
     $  RLKDLWS = RLKDLWS - amin1(runoff_south(is7bsn-6)*86400.
     $  ,RLKDLWS)
C
      IF (ISTAOPT(IS7BSN-6) .EQ. 1) THEN
        UNTRTRF = EXCNNR + kflo(ifls150)
        UNTRTRF_S7 = EXCNNR
      ELSE
        if ((iconvey_canal_no(is150bws-6) .eq.
     $        iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $        (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
     $        .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
     $           then
          UNTRTRF = KFLO(IFLS7)
          UNTRTRF_S7 = UNTRTRF
        else
          UNTRTRF = KFLO(IFLS7)+KFLO(IFLS150)
          UNTRTRF_S7 = KFLO(IFLS7)
        endif
```

```
      ENDIF
C
      IF (iwsbkpwsta .eq. 0 .or. istaopt(is7bsn-6) .eq. 0) THEN
       IF(nint(RLKDLWS/86400.).GT.0) THEN
         RLKDLCFS = RLKDLWS/86400.
         IF(UNTRTRF .GT. RLKDLCFS) THEN
            IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
              IF (LWSBKP_floor(is7bsn-6) .or. LWSLO_floor(iwca2a))
     $         THEN
               ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
              ELSE
               ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
              ENDIF
              ws_bkp_frac_runff = 1.0
               KFLO(IFLS2) = - amin1((UNTRTRF - RLKDLCFS)*ws_bkp_frac
     $          *ws_bkp_frac_runff,rmax_capac_ws_bkp(is7bsn-6))
               KFLO(IFLS7) = KFLO(IFLS7) - amin1((UNTRTRF_S7
     $          - dmnd_lec_sa2_s7)*ws_bkp_frac*ws_bkp_frac_runff
     $          ,rmax_capac_ws_bkp(is7bsn-6))
               ws_bkp_s2_s7 = amin1((UNTRTRF_S7-dmnd_lec_sa2_s7)
     $         *ws_bkp_frac*ws_bkp_frac_runff,rmax_capac_ws_bkp
     $         (is7bsn-6))
               KFLO(IFLS150) = KFLO(IFLS150) - amin1((UNTRTRF
     $             - UNTRTRF_S7)*ws_bkp_frac*ws_bkp_frac_runff
     $             ,rmax_capac_ws_bkp(is7bsn-6)-ws_bkp_s2_s7)
            ENDIF
            RLKDLCFS = 0.0
         ELSE
            if ((iconvey_canal_no(is150bws-6) .eq.
     $      iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $      (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
     $      .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
     $         then
              RLKDLCFS = RLKDLCFS - UNTRTRF
            else
              dmnd_lec_sa3_s150 = dmnd_lec_sa3_s150
     $          - kflo(ifls150)
              RLKDLCFS = RLKDLCFS - UNTRTRF
            endif
         ENDIF
         RLKDLWS = RLKDLCFS*86400.
      ELSE
C
         RLKDLCFS = 0.0
         IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
            IF (LWSBKP_floor(is7bsn-6) .or. LWSLO_floor(iwca2a)) THEN
              ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
            ELSE
              ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
            ENDIF
            ws_bkp_frac_runff = 1.0
            KFLO(IFLS2) = - amin1(UNTRTRF *ws_bkp_frac
     $       *ws_bkp_frac_runff,rmax_capac_ws_bkp(is7bsn-6))
            KFLO(IFLS7) = KFLO(IFLS7) - amin1(UNTRTRF_S7
     $       *ws_bkp_frac*ws_bkp_frac_runff
     $       ,rmax_capac_ws_bkp(is7bsn-6))
            ws_bkp_s2_s7 = amin1(UNTRTRF_S7
```

```
     $         *ws_bkp_frac*ws_bkp_frac_runff,rmax_capac_ws_bkp
     $         (is7bsn-6))
               KFLO(IFLS150) = KFLO(IFLS150) - amin1((UNTRTRF
     $             - UNTRTRF_S7)*ws_bkp_frac*ws_bkp_frac_runff
     $             ,rmax_capac_ws_bkp(is7bsn-6)-ws_bkp_s2_s7)
          ENDIF
        ENDIF
       ENDIF
C
       IF(nint(RLKDLWS/86400.).GT.0) THEN
          LWSWCA = .TRUE.
          if (runoff_south(is7bsn-6) .le. dmnd_lec_sa2
     $         .or. istaopt(is7bsn-6) .eq. 0) then
            WSDOWN = RLKDLWS/86400.
            GFLK2AD = AMIN1(GRFLK2A*ratio_cap_desgn,WSDOWN,s351mx
     $                            - amax1(kflo(ifls2),0.0))
            if (iconvey_c .ne. 0) then
              if (nint(WSDOWN) .gt. 0) then
                lec_dmnd_on_lok(iwca2a,iconvey_c) = 1
              else
                lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
              endif
            endif
          else
            GFLK2AD = 0.0
            if (iconvey_c .ne. 0) then
              lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
            endif
          endif
       ELSE
          LWSWCA = .FALSE.
          WSDOWN = 0.0
          GFLK2AD = 0.0
          if (iconvey_c .ne. 0) then
            lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
          endif
       ENDIF
       wsdown_s7 = amax1(dmnd_lec_sa2_s7 - runoff_south(is7bsn-6)
     $                 ,0.0)
       flow_through_ws_3a = gflk2ad
      ELSE
       WSDOWN = amax1(RLKDLWS/86400.- runoff_south(is7bsn-6)
     $                      ,0.0)
       if (ntarg .gt. 0.and. nnrctwca3a .eq. 'TRUE ') then
         LWSWCA = .TRUE.
         GFLK2AD = AMIN1(GRFLK2A*ratio_cap_desgn,
     $                  s351mx-amax1(kflo(ifls2),0.0))
         if (iconvey_c .ne. 0) then
           if (runoff_south(is7bsn-6) .lt. dmnd_lec_sa2) then
             lec_dmnd_on_lok(iwca2a,iconvey_c) = 1
           else
             lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
           endif
         endif
       else
         GFLK2AD = AMIN1(GRFLK2A*ratio_cap_desgn,WSDOWN,s351mx
     $                          - amax1(kflo(ifls2),0.0))
```

```
            if (nint(WSDOWN) .eq. 0) then
              LWSWCA = .FALSE.
              if (iconvey_c .ne. 0) then
                lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
              endif
            else
              LWSWCA = .TRUE.
              if (iconvey_c .ne. 0) then
                lec_dmnd_on_lok(iwca2a,iconvey_c) = 1
              endif
            endif
          endif
          wsdown_s7 = amax1(dmnd_lec_sa2_s7 - runoff_south(is7bsn-6)
     $                 ,0.0)
          flow_through_ws_3a = gflk2ad
c
C DETERMINE THE AMOUNT OF RUNOFF IN NNR CANAL BASIN IN THE EAA TO BE
C BACKPUMPED TO LAKE OKEECHOBEE FOR WATER SUPPLY.  WILL BE DONE ONLY
C IF LAKE OKEECHOBEE LEVEL IS LOW AND THERE IS NO W.S. OR ENVIRONMENTAL
C NEEDS IN THE WCA'S OR LOWER EAST COAST
            IF (istaopt(is7bsn-6) .eq. 1 .AND. iwsbkpwsta .EQ. 1) THEN
C
             IF (LWSBKP_floor(is7bsn-6) .or. LWSLO_floor(iwca2a)) THEN
                 ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
             ELSE
                 ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
             ENDIF
             ws_bkp_frac_runff = 1.0
             IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
               IF (.NOT. LWSWCA) THEN
                 KFLO(IFLS2) = - amin1(runoff_south(is7bsn-6)*ws_bkp_frac
     $            *ws_bkp_frac_runff,rmax_capac_ws_bkp(is7bsn-6))
                 runoff_south(is7bsn-6) = runoff_south(is7bsn-6) -
     $            amin1(UNTRTRF_S7*ws_bkp_frac*ws_bkp_frac_runff
     $            ,rmax_capac_ws_bkp(is7bsn-6))
               ENDIF
             ENDIF
            ENDIF
c
c    route runoff + flow-through (flow-through may exceed lec demands)
c    to stas if runoff exceeds lec demands
c
             if (istaopt(is7bsn-6) .eq. 1) then
              runoff_s7bsn = runoff_south(is7bsn-6)
               call flow_to_stas(runoff_s7bsn,dmnd_lec_sa2
     $ ,GFLK2AD,2,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $ ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $ ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $ ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $ ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $ ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $ ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $ ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $ ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
             endif
C
            IF (ISTAOPT(IS7BSN-6) .EQ. 1) THEN
```

```
              UNTRTRF = EXCNNR + kflo(ifls150)
              UNTRTRF_S7 = EXCNNR
            ELSE
              if ((iconvey_canal_no(is150bws-6) .eq.
     $          iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $          (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
     $          .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
     $             then
                 UNTRTRF = KFLO(IFLS7)
                 UNTRTRF_S7 = UNTRTRF
              else
                 UNTRTRF = KFLO(IFLS7)+KFLO(IFLS150)
                 UNTRTRF_S7 = KFLO(IFLS7)
              endif
            ENDIF
c
C DETERMINE THE AMOUNT OF RUNOFF IN NNR CANAL BASIN IN THE EAA TO BE
C BACKPUMPED TO LAKE OKEECHOBEE FOR WATER SUPPLY.  WILL BE DONE ONLY
C IF LAKE OKEECHOBEE LEVEL IS LOW AND THERE IS NO W.S. OR ENVIRONMENTAL
C NEEDS IN THE WCA'S OR LOWER EAST COAST
C
             IF (LWSBKP_floor(is7bsn-6) .or. LWSLO_floor(iwca2a)) THEN
                 ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
             ELSE
                 ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
             ENDIF
             ws_bkp_frac_runff = 1.0
             IF(LWSBPMP.AND.KBFBPS2.EQ.0. .AND. (iwsbkpwsta .EQ. 0
     $        .or. istaopt(is7bsn-6) .eq.0)) THEN
              IF (.NOT. LWSWCA) THEN
                KFLO(IFLS2) = - amin1(UNTRTRF *ws_bkp_frac
     $           *ws_bkp_frac_runff,rmax_capac_ws_bkp(is7bsn-6))
                KFLO(IFLS7) = KFLO(IFLS7) - amin1(UNTRTRF_S7
     $           *ws_bkp_frac*ws_bkp_frac_runff
     $           ,rmax_capac_ws_bkp(is7bsn-6))
                ws_bkp_s2_s7 = amin1(UNTRTRF_S7
     $          *ws_bkp_frac*ws_bkp_frac_runff,rmax_capac_ws_bkp
     $          (is7bsn-6))
                KFLO(IFLS150) = KFLO(IFLS150) - amin1((UNTRTRF
     $              - UNTRTRF_S7)*ws_bkp_frac*ws_bkp_frac_runff
     $              ,rmax_capac_ws_bkp(is7bsn-6)-ws_bkp_s2_s7)
              ENDIF
             ENDIF
         ENDIF
C
       ELSE
          LWSWCA = .FALSE.
          GFLK2AD = 0.0
          if (iconv_use_inj(is7bsn-6) .eq. 0) then
            KFLOREF(is7bsn-6) = runoff_south(is7bsn-6)
          else
            KFLOREF(is7bsn-6) = runoff_south_bef_divers(is7bsn-6)
          endif
          flow_through_ws_3a = gflk2ad
          ratio_cap_desgn = 1.0
          if (iconvey_c .ne. 0) then
            lec_dmnd_on_lok(iwca2a,iconvey_c) = 0
```

```
         endif
c
c route runoff into sta34
c
      if (istaopt(is7bsn-6) .eq. 1 ) then
          runoff_s7bsn = runoff_south(is7bsn-6)
          tmpvar = 0.0
          call flow_to_stas(runoff_s7bsn,0.0,tmpvar
     $ ,2,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $ ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $ ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $ ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $ ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $ ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $ ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $ ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $ ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
      endif
      ENDIF
      if (wcaenv(idn_wca_indx_env(is7bsn-6)) .EQ. 'TRUE '.and.
     $        nnrctwca3a .eq. 'TRUE ') THEN
         add_inflow = flow_through_ws_3a
      else
         add_inflow = 0.0
      endif
      wslk2a = GFLK2AD*frac_del_frm_wca(iwca2a)
c
      IF (iconvey_canal_no(is150bws-6) .ne. iconvey_canal_no
     $ (is8bsn-6)) then
         indx_wca = iwca3a
         rmin_stg_wca0 = rmin_stg_wca3a
         ieaa_bsn0 = is8bsn-6
      ELSE
         indx_wca = iwca2a
         rmin_stg_wca0 = rmin_stg_wca2a
         ieaa_bsn0 = is7bsn-6
      ENDIF
      if ((istaopt(is7bsn-6) .eq. 0.or.runoff_south(is7bsn-6)
     $ +add_inflow.le.dmnd_lec_sa2).and.flow_to_wca_prior
     $ .ne. 'LEC' .and. ((((.not. LWSLO_floor(indx_wca) .or. .not.
     $ LWSLECSTA(ieaa_bsn0)) .or. floor_grid_cell_opt.ne.'MARSH')
     $ .AND. rmin_stg_wca0 .le. rmnstg(indx_wca,iconvey_canal_no_wca
     $ (indx_wca)))).or. iopt_ws_lec_to_sta_wca_abv_flr .ne. 1) then
c     if ((runoff_south(is7bsn-6)+add_inflow .le. dmnd_lec_sa2
c    $      .or. istaopt(is7bsn-6) .eq. 0) .and. flow_to_wca_prior
c    $      .ne. 'LEC') then
        if (nint(WSDOWN) .gt. 0) then
          ratio_wsdel_nnrc = amin1(gflk2ad/WSDOWN,1.0)
        else
          ratio_wsdel_nnrc = 1.0
        endif
        if (istaopt(is7bsn-6) .eq. 0) then
        if (eaa_conv_opt_s7 .ne. 'PUMP') then
         if (kflo(ifls7) .le. s7_cap) then
           if ((iconvey_canal_no(is150bws-6) .eq.
     $         iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $         (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
```

```
$         .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
$            then
       rem_sply_cap = amax1(s7_cap - kflo(ifls7),0.0)
       WSLK2A = AMIN1(GFLK2AD,rem_sply_cap)*frac_del_frm_wca
$      (iwca2a)
       rem_cap_sply_s7 = amax1(s7_cap - wslk2a - kflo(ifls7),0.0)
       s7_pump_flow = 0.0
       s7_grav_flow = kflo(ifls7) + wslk2a
       WSLK2A_S7 = WSLK2A
       WSLK2A_S150 = 0.0
     else
       rem_sply_cap = amax1(s7_cap - kflo(ifls7),0.0)
       WSLK2A_S7 = amin1(ratio_wsdel_nnrc*wsdown_s7,rem_sply_cap)
$      *frac_del_frm_wca(iwca2a)
       rem_cap_sply_s7 = amax1(s7_cap-wslk2a_s7-kflo(ifls7),0.0)
       s7_pump_flow = 0.0
       s7_grav_flow = kflo(ifls7) + wslk2a_s7
       WSLK2A_S150 = (GFLK2AD-WSLK2A_S7)*frac_del_frm_wca(iwca3a)
     endif
   else
     WSLK2A = GFLK2AD*frac_del_frm_wca(iwca2a)
     rem_cap_sply_s7 = 0.0
     total_flow_nnr = kflo(ifls7) + wslk2a + kflo(ifls150)
     if (.not. LFORPUMP(is8bsn-6)) then
       cutw = -1.0
       call eaa_prof_lu_table(total_flow_nnr,12.5
$      ,cutw,2,ifacteaa)
       cutw = amax1(cutw,pmp_int(is7bsn-6))
       s351_tw     = 12.5
       s7_spill_hw = cutw
       s7_cap  =  s7max
       s351cap = -1.0
       call eaaspill(s351cap,stagelo,s351_tw,6,ifacteaa)
       nnrc_cap_upper_grav = -1.0
       if (s351_tw.gt.pmp_int(is7bsn-6)) then
          call eaa_prof_lu_table(nnrc_cap_upper_grav,
$         s351_tw,cutw,2,ifacteaa)
       else
          nnrc_cap_upper_grav = 0.0
       endif
     else
       cutw = pmp_int(is7bsn-6)
       rmax_s351_tw = amin1(rmaxstg(is7bsn-6),stagelo
$      + rmax_lift(is7bsn-6))
       s351cap = -1.0
       call eaaspill (s351cap,stagelo,rmax_s351_tw,10,ifacteaa)
       nnrc_cap_upper_grav = s351cap
       s351_tw = -1.0
       call eaa_prof_lu_table(nnrc_cap_upper_grav,s351_tw
$      ,cutw,2,ifacteaa)
       if (s351_tw .gt. rmax_s351_tw) then
          s351_tw = rmax_s351_tw
          nnrc_cap_upper_grav = -1.0
          call eaa_prof_lu_table(nnrc_cap_upper_grav,s351_tw
$         ,cutw,2,ifacteaa)
       endif
       s7_spill_hw = cutw
```

```
        s7_cap  =  s7max
      endif
      fllks6 = -1.0
      if (s351_tw.gt.s6_hw) then
         call eaa_prof_lu_table(fllks6,s351_tw,s6_hw,3,ifacteaa)
      else
         fllks6 = 0.0
      endif
      if ((iconvey_canal_no(is150bws-6) .eq.
$          iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
$          (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
$          .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
$             then
        s7_pump_flow = kflo(ifls7) + wslk2a_s7
        s7_grav_flow = 0.0
        WSLK2A_S150 = 0.0
        WSLK2A_S7 = WSLK2A
      else
        WSLK2A_S7 = amin1(ratio_wsdel_nnrc*wsdown_s7,rem_sply_cap)
$       * frac_del_frm_wca(iwca2a)
        s7_pump_flow = kflo(ifls7) + wslk2a_s7
        s7_grav_flow = 0.0
        WSLK2A_S150 = (gflk2ad - WSLK2A_S7)*frac_del_frm_wca(iwca3a)
      endif
    endif
   else
    if (igravity_s7_nosta .eq. 1) then
      if ((iconvey_canal_no(is150bws-6) .eq.
$          iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
$          (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
$          .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
$             then
        rem_sply_cap = amax1(s7_cap - kflo(ifls7) ,0.0)
        WSLK2A_S7 = AMIN1(GFLK2AD,rem_sply_cap)
$       * frac_del_frm_wca(iwca2a)
        WSLK2A = WSLK2A_S7
        WSLK2A_S150 = 0.0
        rem_cap_sply_s7 = amax1(s7_cap-wslk2a_s7-kflo(ifls7),0.0)
        s7_pump_flow = 0.0
        s7_grav_flow = kflo(ifls7) + wslk2a_s7
      else
        WSLK2A_S7 = amin1(ratio_wsdel_nnrc*wsdown_s7,rem_sply_cap)
$       *frac_del_frm_wca(iwca2a)
        WSLK2A_S150 = (GFLK2AD - WSLK2A_S7)*frac_del_frm_wca(iwca3a)
        rem_cap_sply_s7 = amax1(s7_cap-wslk2a_s7-kflo(ifls7),0.0)
        s7_pump_flow = 0.0
        s7_grav_flow = kflo(ifls7) + wslk2a_s7
      endif
    else
      s7_spill_hw = pmp_int(is7bsn-6)
      s7_cap  =  s7max
      if ((iconvey_canal_no(is150bws-6) .eq.
$          iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
$          (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
$          .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
$             then
        WSLK2A = GFLK2AD*frac_del_frm_wca(iwca2a)
```

```
        WSLK2A_S7 = WSLK2A
        WSLK2A_S150 = 0.0
        rem_cap_sply_s7 = 0.0
        s7_pump_flow = kflo(ifls7) + wslk2a_s7
        s7_grav_flow = 0.0
        if (IPRORTE(IWCA3A) .EQ. 1) THEN
          if (nint(dmnd_lec_sa3) .gt.0) then
            prorate_ratio = wslk2a_s150 / (dmnd_lec_sa3
$               + rlocal_storage_3a)
            prorate_ratio = amin1(prorate_ratio,1.0)
          else
            prorate_ratio = 0.0
          endif
          rlok_flow_to_enp_s150 = prorate_ratio * frf
          rlok_flow_to_enp = rlok_flow_to_enp_s150
$                 + rlok_flow_to_enp_s354
        endif
      else
        WSLK2A = GFLK2AD*frac_del_frm_wca(iwca2a)
        WSLK2A_S7 = ratio_wsdel_nnrc*wsdown_s7
        WSLK2A_S150 = (GFLK2AD - WSLK2A_S7)*frac_del_frm_wca(iwca3a)
        if (IPRORTE(IWCA3A) .EQ. 1) THEN
          if (nint(dmnd_lec_sa3_s150) .gt.0) then
            prorate_ratio = wslk2a_s150 / (dmnd_lec_sa3_s150
$               + rlocal_storage_3a)
            prorate_ratio = amin1(prorate_ratio,1.0)
          else
            prorate_ratio = 0.0
          endif
          rlok_flow_to_enp_s150 = prorate_ratio * frf
          rlok_flow_to_enp = rlok_flow_to_enp_s150
$                 + rlok_flow_to_enp_s354
        endif
        rem_cap_sply_s7 = 0.0
        s7_pump_flow = kflo(ifls7) + wslk2a_s7
        s7_grav_flow = 0.0
      endif
    endif
  endif
  else
    if ((iconvey_canal_no(is150bws-6) .eq.
$         iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
$         (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
$         .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
$            then
      if (kflo(ifls7) .gt. s7_cap) then
        wslk2a_s7 = 0.0
        wslk2a_s150 = 0.0
        rem_cap_sply_s7 = 0.0
      else
        wslk2a = amin1(gflk2ad,s7_cap-kflo(ifls7))
$       * frac_del_frm_wca(iwca2a)
        wslk2a_s7 = wslk2a
        rem_cap_sply_s7 = amax1(s7_cap - wslk2a - kflo(ifls7),0.0)
        wslk2a_s150 = 0.0
      endif
```

```
            if (IPRORTE(IWCA3A) .EQ. 1) THEN
              if (nint(dmnd_lec_sa3) .gt.0) then
                prorate_ratio = wslk2a_s150 / (dmnd_lec_sa3
     $                + rlocal_storage_3a)
                prorate_ratio = amin1(prorate_ratio,1.0)
              else
                prorate_ratio = 0.0
              endif
              rlok_flow_to_enp_s150 = prorate_ratio * frf
              rlok_flow_to_enp = rlok_flow_to_enp_s150
     $                 + rlok_flow_to_enp_s354
            endif
          else
           if (kflo(ifls7) .gt. s7_cap) then
             wslk2a_s7 = 0.0
             wslk2a_s150 = wslk2a
             rem_cap_sply_s7 = 0.0
           else
             wslk2a_s7=amin1(ratio_wsdel_nnrc*wsdown_s7,s7_cap
     $               -kflo(ifls7))*frac_del_frm_wca(iwca2a)
             rem_cap_sply_s7 = amax1(s7_cap -wslk2a_s7-kflo(ifls7),0.0)
             wslk2a_s150 = (wslk2a - wslk2a_s7)*frac_del_frm_wca(iwca3a)
           endif
            if (IPRORTE(IWCA3A) .EQ. 1) THEN
              if (nint(dmnd_lec_sa3_s150) .gt.0) then
                prorate_ratio = wslk2a_s150 / (dmnd_lec_sa3_s150
     $               + rlocal_storage_3a)
                prorate_ratio = amin1(prorate_ratio,1.0)
              else
                prorate_ratio = 0.0
              endif
              rlok_flow_to_enp_s150 = prorate_ratio * frf
              rlok_flow_to_enp = rlok_flow_to_enp_s150
     $                 + rlok_flow_to_enp_s354
            endif
c           print *,kflo(ifls7),s7_cap,ratio_wsdel_nnrc
c    $          ,wsdown_s7,wslk2a_s150,gflk2ad,grflk2a
c    $          ,'  WSLKEA GRP ',ratio_cap_desgn
         endif
        endif
        flow_s7_tmp = kflo(ifls7)
        flow_s150_tmp = kflo(ifls150)
c
        if (ntarg_hill .gt. 0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1 .or.
     $         ires_opt_hill .eq. 1)) then
          flow_s7_tmp = flow_s7_tmp + WSLK2A_S7
          flow_s2_tmp = flow_s2_tmp + wslk2a
          flow_s150_tmp = flow_s150_tmp + wslk2a_s150
          if (conv_asr .eq. 1) then
             kflo(ifls7) = flow_s7_tmp
             kflo(ifls2) = flow_s2_tmp
             kflo(ifls150) = flow_s150_tmp
             DELLEC = DELLEC + WSLK2A * 1.9835
             WSS351 = WSS351 + WSLK2A
             kfloref(is7bsn-6) = kfloref(is7bsn-6) + wslk2a
             ws_to_lec_s351 = ws_to_lec_s351 + amax1(WSLK2A
     $       - rlok_flow_to_enp_s150,0.0)
```

```
            ws_to_lec_s351_nnrc = wslk2a_s7 + ws_to_lec_s351_nnrc
            ws_to_lec_s351_wca3a = ws_to_lec_s351_wca3a
     $           + amax1(wslk2a_s150 - rlok_flow_to_enp_s150,0.0)
            if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
              ws_pipeline_3a_s150 = ws_pipeline_3a_s150 + wslk2a_s150
     $                - rlok_flow_to_enp_s150
            endif
            if (opt_pipeline_ws(is7bsn-6) .eq. 'PIPEL  ') then
              ws_pipeline_2a = ws_pipeline_2a + wslk2a_s7
            endif
          endif
        else
          KFLO(IFLS7) = KFLO(IFLS7) + WSLK2A_S7
          if (opt_pipeline_ws(is7bsn-6) .eq. 'PIPEL  ') then
            ws_pipeline_2a = ws_pipeline_2a + WSLK2A_S7
          endif
          KFLO(IFLS2) = KFLO(IFLS2) + WSLK2A
          KFLO(IFLS150) = KFLO(IFLS150) + WSLK2A_S150
          kfloref(is7bsn-6) = kfloref(is7bsn-6) + WSLK2A
          if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
            ws_pipeline_3a_s150 = ws_pipeline_3a_s150 + wslk2a_s150
          endif
c         print *,wslk2a_s150,KFLO(IFLS150),'  S150  ',
c    $        WSLK2A_S7,WSLK2A
          flow_s7_tmp = KFLO(IFLS7)
          flow_s2_tmp = kflo(ifls2)
          flow_s150_tmp = KFLO(IFLS150)
          DELLEC = DELLEC + WSLK2A * 1.9835
          WSS351 = WSS351 + WSLK2A
          ws_to_lec_s351 = ws_to_lec_s351 + amax1(WSLK2A
     $    - rlok_flow_to_enp_s150,0.0)
          ws_to_lec_s351_nnrc = wslk2a_s7 + ws_to_lec_s351_nnrc
          ws_to_lec_s351_wca3a = ws_to_lec_s351_wca3a
     $          + amax1(wslk2a_s150 - rlok_flow_to_enp_s150,0.0)
        endif
c       if (opt_pipeline_ws(is7bsn-6) .eq. 'PIPEL  ') then
c         add_ws_pipeline_2a = amin1(kflo(ifls7),runoff_south(is7bsn-6)
c    $         ,dmnd_lec_sa2_s7)
c         ws_pipeline_2a = ws_pipeline_2a +  add_ws_pipeline_2a
c       endif
        if (ntarg_hill .gt. 0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1 .or.
     $    ires_opt_hill .eq. 1) .and. conv_asr.eq.1 .or.
     $    ((iasr_opt_eaa(is7bsn-6) .ne. 1 .and. ires_opt_hill
     $   .ne. 1) .or. ntarg_hill .eq. 0)) then
          if (istaopt(is7bsn-6) .eq. 1) then
            if ( eaa_conv_opt_s7 .eq. 'PUMP') then
              if (igravity_s7 .eq. 1) then
                s7_pump_flow = 0.0
                s7_grav_flow = kflo(ifls7)
              else
                s7_grav_flow = 0.0
                s7_pump_flow = kflo(ifls7)
              endif
            else
              s7_pump_flow = amax1(kflo(ifls7) - s7_cap,0.0)
              s7_grav_flow = amin1(kflo(ifls7),s7_cap)
            endif
```

```
          endif
        endif
       endif
        if (ntarg_hill .gt. 0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1 .or.
     $    ires_opt_hill .eq. 1) .and. conv_asr.eq.1 .or.
     $    ((iasr_opt_eaa(is7bsn-6) .ne. 1 .and. ires_opt_hill
     $   .ne. 1) .or. ntarg_hill .eq. 0)) then
        avvol_upstream = -901.
        str_name = 's351n '
        cname_lower = 'nnrclw'
        cname_upper = 'nnrcup'
        str_name_new_pmp = 'g370  '
        str_name_new = 's7newg'
        if (istaopt(is7bsn-6) .eq. 1) then
         if (eaa_conv_opt_s7 .eq. 'PUMP' .or. runoff_south(is7bsn-6)
     $         .gt. avail_cap_for_sta34_nnr) then
           nnrc_cap       = nnrc_cap_upper_pump
           nnrc_cap_lower = nnrc_cap_lower_pump
           if (igravity_s7 .eq. 1) then
             s7_pump_flow = 0.0
             s7_grav_flow = kflo(ifls7)
           else
             s7_grav_flow = 0.0
             s7_pump_flow = kflo(ifls7)
           endif
         else
           nnrc_cap       = nnrc_cap_upper_grav
           nnrc_cap_lower = nnrc_cap_lower_grav
           s7_pump_flow = 0.0
           s7_grav_flow = kflo(ifls7)
         endif
        else
         nnrc_cap       = nnrc_cap_upper_grav
         nnrc_cap_lower = nnrc_cap_lower_grav
        endif
c
c  water supply thru s-150 into wca-3a to help meet needs
c  in sevice area 3. delivery made only if not interfere
c  with flow thru s-7 into wca-2a
c
       if (flow_to_wca_prior .ne. 'LEC') then
        if ((iconvey_canal_no(is150bws-6) .eq.
     $         iconvey_canal_no(is8bsn-6) .and. idn_wca_indx
     $         (is150bws-6) .eq. idn_wca_indx(is8bsn-6))
     $         .and. opt_pipeline_ws(is150bws-6) .ne. 'PIPEL  ')
     $            then
          rem_ws_needs_3a = amax1(dmnd_lec_sa3 - kflo(ifls8)
     $                - ws_pipeline_3a,0.0)
        else
          rem_ws_needs_3a = 0.0
        endif
        s150cap = -1.0
        call eaaspill(s150cap,s7_spill_hw,s150_tailw,8,ifacteaa)
        rem_cap_s150 = amax1(s150cap-kflo(ifls150),0.0)
         if (istaopt(is7bsn-6) .eq. 1) then
           rmax_thru_s150 = amin1(amax1(grflk2a*ratio_cap_desgn-wslk2a
     $       ,0.0),amax1(flow_lower_reach_max - kflo(ifls7)
```

```
$      + dischg_to_s7_frm_sta34 - wslk2a,0.0))
   else
     rmax_thru_s150 = amax1(grflk2a*ratio_cap_desgn-wslk2a,0.0)
   endif
   flow_through_max = amin1(rmax_thru_s150,
$        s351cap-amax1(kflo(ifls2),0.0)
$        ,rem_cap_s150)
   flow_through_max = amax1(flow_through_max,0.0)
   flow_through_max = amin1(flow_through_max,rem_ws_needs_3a)
$    * frac_del_frm_wca(iwca3a)
   if (IPRORTE(IWCA3A) .EQ. 1) THEN
      if (nint(dmnd_lec_sa3) .gt.0) then
        prorate_ratio = flow_through_max / (dmnd_lec_sa3
$               + rlocal_storage_3a)
        prorate_ratio = amin1(prorate_ratio,1.0)
      else
        prorate_ratio = 0.0
      endif
      rlok_flow_to_enp_s150 = prorate_ratio * frf
      rlok_flow_to_enp = rlok_flow_to_enp_s150
$               + rlok_flow_to_enp_s354
    endif
  if (ntarg_hill.gt.0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1
$   .or. ires_opt_hill .eq. 1)) then
    flow_s2_tmp   = flow_s2_tmp  + flow_through_max
    flow_s150_tmp = kflo(ifls150) + flow_through_max
    if (conv_asr .eq. 1) then
      kflo(ifls2) = flow_s2_tmp
      kfloref(is7bsn-6) = kfloref(is7bsn-6) + flow_through_max
      kflo(ifls150) = flow_s150_tmp
      wss351 = wss351 + flow_through_max
      ws_to_lec_s351 = ws_to_lec_s351 + amax1(flow_through_max
$      - rlok_flow_to_enp_s150,0.0)
      ws_to_lec_s351_wca3a = flow_through_max
$        + amax1(ws_to_lec_s351_wca3a-rlok_flow_to_enp_s150,0.0)
      dellec = dellec + flow_through_max * 1.9835
      if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
        ws_pipeline_3a_s150 =  ws_pipeline_3a_s150
$               + flow_through_max
      endif
    endif
  else
    kflo(ifls2) = kflo(ifls2) + flow_through_max
    kfloref(is7bsn-6) = kfloref(is7bsn-6) + flow_through_max
    wss351 = wss351 + flow_through_max
    ws_to_lec_s351 = ws_to_lec_s351 + amax1(flow_through_max
$   - rlok_flow_to_enp_s150,0.0)
    ws_to_lec_s351_wca3a = flow_through_max
$       + amax1(ws_to_lec_s351_wca3a-rlok_flow_to_enp_s150,0.0)
    dellec = dellec + flow_through_max * 1.9835
    kflo(ifls150) = kflo(ifls150) + flow_through_max
    if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
      ws_pipeline_3a_s150 =  ws_pipeline_3a_s150
$           + flow_through_max
    endif
  endif
  s150_wsdel = flow_through_max
```

```
      if (eaa_conv_opt_s7 .eq. 'PUMP') then
        rnnrc_cap_neutral = eaa_convey(5,1)
      else
        rnnrc_cap_neutral = eaa_convey(3,1)
      endif
      if (ntarg_hill.gt.0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1
     $   .or. ires_opt_hill .eq. 1)) then
        if (nnrc_cap .lt. rnnrc_cap_neutral) then
          nnrc_cap_temp = nnrc_cap + s150_wsdel
          nnrc_cap      = nnrc_cap_temp
        endif
      else
        if (nnrc_cap .lt. rnnrc_cap_neutral) then
          nnrc_cap = nnrc_cap + s150_wsdel
        endif
      endif
     endif
c
      if (ntarg_hill .gt. 0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1 .or.
     $    ires_opt_hill .eq. 1) .and. conv_asr.eq.1 .or.
     $    ((iasr_opt_eaa(is7bsn-6) .ne. 1 .and. ires_opt_hill
     $   .ne. 1) .or. ntarg_hill .eq. 0)) then
     total_flow_s2 = wss351 + dmnd_nnr
     if (lp(23)) then
      write(62,235) ifyr+iyear-1,month,iday,str_name,stagelo
     $  ,s351_tw,s351cap,avvol_upstream,total_flow_s2,dmnd_lec_sa2
     $  +rem_ws_needs_3a
     endif
c
     if (istaopt(is7bsn-6) .eq. 0) s7_new_hw = s7_spill_hw
     if (iconv_red_nnrc .eq. 0) then
       total_flow_upper_reach = wss351+runoff_south(is7bsn-6)
     $               + dmnd_nnr
     else
       total_flow_upper_reach = wss351+runoff_south(is7bsn-6)
     $        + dmnd_nnr + dmnd_nnr_orig - dmnd_nnr_aft_resasr
     endif
     if (lp(23)) then
     write(62,235) ifyr+iyear-1,month,iday,cname_upper,s351_tw
     $  ,s7_new_hw,amax1(nnrc_cap,0.0)
     $  ,avvol_upstream,total_flow_upper_reach,dmnd_lec_sa2
     $  +rem_ws_needs_3a
     if ( istaopt(is7bsn-6) .eq. 1) then
       wsneeds_downstrm = 0.0
       write(62,235) ifyr+iyear-1,month,iday,str_name_new_pmp,s7_new_hw
     $ ,chdep(ista34),g370cap,avvol_upstream,flow_s7new_to_sta
     $ ,wsneeds_downstrm
       flow_total = kflo(ifls7) + flow_through_max
       write(62,235) ifyr+iyear-1,month,iday,str_name_new,s7_new_hw
     $ ,s7_new_tw,s7_cap_new,avvol_upstream,flow_total
     $ -dischg_to_s7_frm_sta34
     $ ,dmnd_lec_sa2+rem_ws_needs_3a
       write(62,235) ifyr+iyear-1,month,iday,cname_lower,s7_new_tw
     $  ,s7_spill_hw,flow_lower_reach_max,avvol_upstream
     $  ,flow_total-dischg_to_s7_frm_sta34,dmnd_lec_sa2
     $  +rem_ws_needs_3a
     endif
```

```
      write(62,235) ifyr+iyear-1,month,iday,flnm(ifls7),s7_spill_hw
     $  ,s7_tailw,s7_cap,avvol_upstream,kflo(ifls7)
     $  ,dmnd_lec_sa2
      write(62,235) ifyr+iyear-1,month,iday,flnm(ifls150),s7_spill_hw
     $  ,s150_tailw,s150cap,avvol_upstream,kflo(ifls150)
     $  ,rem_ws_needs_3a
      endif
      endif
      endif
      if (opt_pipeline_ws(is7bsn-6) .eq. 'PIPEL  ') then
         kflo(ifls7) = kflo(ifls7) - ws_pipeline_2a
      endif
      if (opt_pipeline_ws(is150bws-6) .eq. 'PIPEL  ') then
         kflo(ifls150) = kflo(ifls150) - ws_pipeline_3a_s150
      endif
C
C TO WCA-1 VIA HILLSBORO CANAL THROUGH S-6
C
C
       if  (wcaenv(idn_wca_indx_env(is6bsn-6)) .eq. 'FALSE') then
c
c route eaa runoff to asr reservoir if there are no LEC demands and
c no environmental demands in everglades
c
        if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_hill
     $       .eq. 1) then
           runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $          - dmnd_lec_s6-wsuply(indxsta(is6bsn-6))
     $          - wsuply(indx_sta2w_frm_lok_ws),0.0)
           res_flow = amin1(runoff_to_inject
     $      *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $      ,rem_weir_capac(is7bsn-6))
           flow_to_eaa_res_rnff(is7bsn-6) = res_flow
     $      + flow_to_eaa_res_rnff(is7bsn-6)
           rem_weir_capac(is7bsn-6) =rem_weir_capac(is7bsn-6)
     $          - res_flow
           volwtr(ieaa_res_asr(is6bsn-6))
     $        = volwtr(ieaa_res_asr(is6bsn-6)) - res_flow
           divers_to_res = amin1(divers,res_flow)
           divers = divers - divers_to_res
           runoff_south(is6bsn-6) = runoff_south(is6bsn-6)
     $         - (res_flow - divers_to_res)
           if (istaopt(is6bsn-6) .eq. 0) then
             q = runoff_south(is6bsn-6)
             IF (Q .GT. S6MAX) THEN
               KFLO(IFLS6) = S6MAX
               IAGFLO(IAG(IS7BSN),1) = NINT(Q - S6MAX)
     &               + IAGFLO(IAG(IS7BSN),1)
             ELSE
               KFLO(IFLS6) = Q
             ENDIF
           endif
        endif
       else
c
c route eaa runoff to asr reservoir if there are no LEC demands and
c no environmental demands in everglades
```

```
c
          if (iasr_opt_eaa(is7bsn-6) .eq. 1 .or. ires_opt_hill
     $        .eq. 1) then
           if (ntarg_hill .eq. 0) then
             runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $         - dmnd_lec_s6-wsuply(indxsta(is6bsn-6))
     $         - wsuply(indx_sta2w_frm_lok_ws),0.0)
           else
             rem_cap_s351_aft = s351mx - flow_s2_tmp
     $               - wsuply(indxsta(is6bsn-6))
             rmax_flow_thru = amin1(rem_cap_s351_aft
     $        ,fllks6 - dmnd_hill - wsuply(indxsta(is6bsn-6))
     $        - wsuply(indx_sta2w_frm_lok_ws),s6max)
             rmax_flow_thru = amax1(rmax_flow_thru,0.0)
             rmax_flow_bypass_res = amax1(rmax_flow_thru,dmnd_lec_s6)
             runoff_to_inject = amax1(runoff_south(is6bsn-6)+divers
     $                - rmax_flow_bypass_res,0.0)
           endif
           res_flow = amin1(runoff_to_inject
     $      *frac_runoff_asr(is7bsn-6),volwtr(ieaa_res_asr(is7bsn-6))
     $      ,rem_weir_capac(is7bsn-6))
           if ((ntarg_hill .gt. 0 .and. conv_asr .eq. 1) .or.
     $         ntarg_hill .eq. 0) then
             flow_to_eaa_res_rnff(is7bsn-6) = res_flow
     $         + flow_to_eaa_res_rnff(is7bsn-6)
             rem_weir_capac(is7bsn-6) =rem_weir_capac(is7bsn-6)
     $           - res_flow
             volwtr(ieaa_res_asr(is7bsn-6))
     $           = volwtr(ieaa_res_asr(is7bsn-6)) - res_flow
           endif
           divers_to_res = amin1(divers,res_flow)
            rem_runoff_south = runoff_south(is6bsn-6)
     $         - (res_flow - divers_to_res)
           if ((ntarg_hill .gt. 0 .and. conv_asr .eq. 1) .or.
     $         ntarg_hill .eq. 0) then
             divers = divers - divers_to_res
             runoff_south(is6bsn-6) = rem_runoff_south
             if (istaopt(is6bsn-6) .eq. 0) then
               q = runoff_south(is6bsn-6)
               IF (Q .GT. S6MAX) THEN
                 KFLO(IFLS6) = S6MAX
                 IAGFLO(IAG(IS7BSN),1) = NINT(Q - S6MAX)
     &              + IAGFLO(IAG(IS7BSN),1)
               ELSE
                 KFLO(IFLS6) = Q
               ENDIF
             endif
           endif
          endif
        endif
        if (kbfbps2 .ge. 0.0) then
         if (conv_asr .ne. 1) then
          if (ntarg_hill .gt. 0 .and. (iasr_opt_eaa(is7bsn-6) .eq. 1
     $      .or. ires_opt_hill .eq. 1)) then
           rem_cap_s351_diff = rem_cap_s351_bef - rem_cap_s351_aft
           if (rem_cap_s351_diff .ne. 0.0) then
             if (rem_cap_s351_diff/rem_cap_s351_diff_prev .lt. 0.0
```

```
     $          .and. iter_asr .ne. 1) then
              flow_incr = abs(flow_incr)*0.5*rem_cap_s351_diff
     $                        /abs(rem_cap_s351_diff)
            endif
          endif
          if (iter_asr.eq.1.and.rem_cap_s351_diff.lt.0.0) flow_incr
     $          =  - flow_incr_init
          if (abs(rem_cap_s351_diff) .lt. 10.0 .or. iter_asr .ge. 25)
     $        then
            rem_cap_s351_diff_prev = rem_cap_s351_diff
            conv_asr = 1.
            flow_incr = 0.0
            go to 777
          else
            tmp_flw_s2 = tmp_flw_s2 + flow_incr
            rem_cap_s351_diff_prev = rem_cap_s351_diff
            iter_asr = iter_asr + 1
            go to 777
          endif
        endif
      endif
     endif

c
      IF(KBFBPS2.GE.0.) THEN
C
c
        if (iconv_use_inj(is6bsn-6) .eq. 0) then
          kfloref(is6bsn-6) = runoff_south(is6bsn-6)+divers
     $       +dmnd_hill
        else
          kfloref(is6bsn-6) = runoff_south_bef_divers(is6bsn-6)
     $       +dmnd_hill
        endif
        add_red_in_conv = 0.0
        do irs = 1,nres_ws
         if (iconv_use_opt_ws(irs) .eq. 1) then
          do js = 1,noutlets(irs)
            if (ieaa_bsn(irs,js) .eq. is7bsn-6) then
              add_red_in_conv = add_red_in_conv +
     $          res_outflow_to_basin(irs,js)
            endif
          enddo
         endif
        enddo
        add_red_in_conv = add_red_in_conv - dmnd_nnr_orig
     $            + dmnd_nnr_aft_resasr
        add_red_in_conv = amax1(add_red_in_conv,0.0)
        kfloref(is6bsn-6) = kfloref(is6bsn-6) + add_red_in_conv
c
c  water supply from LOK to STA-2
c
        if (istaopt(is6bsn-6) .eq. 1) then
           wsdel_to_sta(indxsta(is6bsn-6)) = amin1(wsuply
     $       (indxsta(is6bsn-6)),amax1(fllks6 - kfloref
     $       (is6bsn-6),0.0),amax1(s351mx-amax1(kflo(ifls2),0.0)
     $       ,0.0))
```

```
          kfloref(is6bsn-6) = kfloref(is6bsn-6)
     $          + wsdel_to_sta(indxsta(is6bsn-6))
          wss351 = wss351 + wsdel_to_sta(indxsta(is6bsn-6))
          kflo(ifls2) = kflo(ifls2) + wsdel_to_sta(indxsta(is6bsn-6))
          wsdel_to_sta(indx_sta2w_frm_lok_ws) = amin1(wsuply
     $     (indx_sta2w_frm_lok_ws),amax1(fllks6 - kfloref
     $     (is6bsn-6),0.0),amax1(s351mx-amax1(kflo(ifls2),0.0)
     $    ,0.0))
          wsuply(indxsta(is6bsn-6)) = wsuply(indxsta(is6bsn-6))
     $      - wsdel_to_sta(indxsta(is6bsn-6))
          wsuply(indx_sta2w_frm_lok_ws)= wsuply(indx_sta2w_frm_lok_ws)
     $      - wsdel_to_sta(indx_sta2w_frm_lok_ws)
          kflo(ifls2)=kflo(ifls2)+wsdel_to_sta(indx_sta2w_frm_lok_ws)
          kfloref(is6bsn-6) = kfloref(is6bsn-6)
     $            + wsdel_to_sta(indx_sta2w_frm_lok_ws)
        endif
        rem_canal_cap_hill = fllks6 - kfloref(is6bsn-6)
        rem_canal_cap_hill = amax1(rem_canal_cap_hill,0.0)
        total_rem_cap_nnrhill = grflk2a + rem_canal_cap_hill
        if (total_rem_cap_nnrhill .gt. 0.0) then
          ratio_cap_desgn = (s351mx-kbfbps2-wsdel_to_sta
     $          (indxsta(is6bsn-6))) / total_rem_cap_nnrhill
          ratio_cap_desgn = amin1(ratio_cap_desgn,1.0)
          ratio_cap_desgn = amax1(ratio_cap_desgn,0.0)
        else
          ratio_cap_desgn = 1.0
        endif
C
      if  (wcaenv(idn_wca_indx_env(is6bsn-6)) .eq. 'FALSE') then
       if (istaopt(is6bsn-6).eq.1 .and. iwsbkpwsta .EQ. 1) THEN
        IF(nint(ws_down_s6).GE.0) THEN
C
          IF(runoff_south(is6bsn-6) .GT. ws_down_s6) THEN
             IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
                IF (LWSBKP_floor(is6bsn-6) .or. LWSLO_floor(iwca1))
     $          THEN
                   ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
                ELSE
                   ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
                ENDIF
                ws_bkp_frac_runff = 1.0
                KFLO(IFLS2) = KFLO(IFLS2) - (runoff_south(is6bsn-6)
     $          - ws_down_s6)*ws_bkp_frac*ws_bkp_frac_runff
                flow_s2_bef_cap_lim = abs(KFLO(IFLS2))
                IF (KFLO(IFLS2) .LT. -0.000001) THEN
                   KFLO(IFLS2) = -amin1(abs(KFLO(IFLS2))
     $             ,rmax_capac_ws_bkp(is7bsn-6))
                ENDIF
                runoff_south(is6bsn-6) = ws_down_s6
     $             + flow_s2_bef_cap_lim - amin1(flow_s2_bef_cap_lim
     $             ,rmax_capac_ws_bkp(is7bsn-6))
     $             + (runoff_south(is6bsn-6) - ws_down_s6)*(1.0
     $             -ws_bkp_frac*ws_bkp_frac_runff)
             ENDIF
          ENDIF
        ENDIF
       endif
```

```
c
c    route runoff from hillsboro canal basin into sta-2
c
         if (istaopt(is6bsn-6) .eq. 1) then
           runoff_s6bsn = runoff_south(is6bsn-6)
           tmpvar = 0.0
           call flow_to_stas (runoff_s6bsn+divers,ws_down_s6/86400.
     $,tmpvar,3,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $,kfloref,runoff_south,flow_from_eaa_lok_s8
     $,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
         endif
c
            IF (ISTAOPT(IS6BSN-6) .EQ. 1) THEN
             UNTRTRF = HILLEXC
            ELSE
             UNTRTRF = KFLO(IFLS6)
            ENDIF
C
         IF (iwsbkpwsta .EQ. 0 .or. istaopt(is6bsn-6) .eq. 0) THEN
          IF(nint(ws_down_s6).GT.0) THEN
C
            IF(UNTRTRF .GT. ws_down_s6) THEN
               IF(LWSBPMP.AND.KBFBPS2.EQ.0.) THEN
                  IF (LWSBKP_floor(is6bsn-6) .or. LWSLO_floor(iwca1))
     $            THEN
                     ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
                  ELSE
                     ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
                  ENDIF
                  ws_bkp_frac_runff = 1.0
                  KFLO(IFLS2) = KFLO(IFLS2) - (UNTRTRF - ws_down_s6)
     $            *ws_bkp_frac*ws_bkp_frac_runff
                  flow_s2_bef_cap_lim = abs(KFLO(IFLS2))
                  IF (KFLO(IFLS2) .LT. -0.000001) THEN
                     KFLO(IFLS2) = -amin1(abs(KFLO(IFLS2))
     $               ,rmax_capac_ws_bkp(is7bsn-6))
                  ENDIF
                  IF (ISTAOPT(IS6BSN-6) .EQ. 0) THEN
                   KFLO(IFLS6) = ws_down_s6 + flow_s2_bef_cap_lim
     $                 - amin1(flow_s2_bef_cap_lim
     $                 ,rmax_capac_ws_bkp(is7bsn-6))
     $                 + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $                 *ws_bkp_frac_runff)
                   untrtrf_south = KFLO(IFLS6)
                  else
                   untrtrf_south = ws_down_s6 + flow_s2_bef_cap_lim
     $                 - amin1(flow_s2_bef_cap_lim
     $                 ,rmax_capac_ws_bkp(is7bsn-6))
     $                 + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $                 *ws_bkp_frac_runff)
                  ENDIF
```

```
            ELSE
              untrtrf_south = untrtrf
            ENDIF
            ws_down_s6 = 0.0
         ELSE
            ws_down_s6 = ws_down_s6 - UNTRTRF
            untrtrf_south = untrtrf
         ENDIF
      ELSE
         IF(LWSBPMP.AND.KBFBPS2.EQ.0) THEN
            IF (LWSBKP_floor(is6bsn-6) .or. LWSLO_floor(iwca1))
     $      THEN
               ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
            ELSE
               ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
            ENDIF
            ws_bkp_frac_runff = 1.0
            KFLO(IFLS2) = KFLO(IFLS2) - UNTRTRF*ws_bkp_frac
     $      *ws_bkp_frac_runff
            flow_s2_bef_cap_lim = abs(KFLO(IFLS2))
            IF (KFLO(IFLS2) .LT. -0.000001) THEN
               KFLO(IFLS2) = -amin1(abs(KFLO(IFLS2))
     $         ,rmax_capac_ws_bkp(is7bsn-6))
            ENDIF
            if (istaopt(is6bsn-6) .eq. 0) then
              KFLO(IFLS6) = flow_s2_bef_cap_lim
     $              - amin1(flow_s2_bef_cap_lim
     $              ,rmax_capac_ws_bkp(is7bsn-6))
     $              + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $              *ws_bkp_frac_runff)
            endif
            untrtrf_south = flow_s2_bef_cap_lim
     $              - amin1(flow_s2_bef_cap_lim
     $              ,rmax_capac_ws_bkp(is7bsn-6))
     $              + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $              *ws_bkp_frac_runff)
         ELSE
          untrtrf_south = untrtrf
         ENDIF
      ENDIF
     ENDIF
C
     IF(nint(RLKDLWS_SA1/86400.).GT.0 ) THEN
        LWSWCA = .TRUE.
        if (runoff_south(is6bsn-6)+divers .le. dmnd_lec_s6 .or.
     $    istaopt(is6bsn-6) .eq. 0) then
          WSDOWN = ws_down_s6
          rem_des_cap = rem_canal_cap_hill*ratio_cap_desgn
          QS6DL = AMIN1(WSDOWN,rem_des_cap,s351mx-amax1(kflo(ifls2)
     $                    ,0.0))
          QS6DL = AMAX1(QS6DL,0.0)*frac_del_frm_wca(iwca1)
        else
          QS6DL = 0.0
        endif
     ELSE
        LWSWCA = .FALSE.
        QS6DL = 0.0
```

```
          ENDIF
          flow_through_s6 = qs6dl
         else
           rem_des_cap = rem_canal_cap_hill*ratio_cap_desgn
           if (ntarg_hill .gt. 0) then
             LWSWCA = .TRUE.
             ws_env_to_wca2a = amin1(rem_des_cap,s351mx-amax1(kflo(ifls2)
     $                   ,0.0))
           else
             wsdown = amax1(ws_down_s6-runoff_south(is6bsn-6)-divers,0.0)
             LWSWCA = .FALSE.
             ws_env_to_wca2a = amin1(rem_des_cap,s351mx-amax1(kflo(ifls2)
     $                  ,0.0),wsdown)*frac_del_frm_wca(iwca1)
           endif
           ws_env_to_wca2a = amax1(ws_env_to_wca2a,0.0)
           flow_through_s6 = ws_env_to_wca2a
           qs6dl = flow_through_s6
cccc
           if (istaopt(is6bsn-6).eq.1 .and. iwsbkpwsta .EQ. 1) THEN
            if (LWSBPMP .and. KBFBPS2 .EQ. 0) THEN
             IF (LWSBKP_floor(is6bsn-6) .or. LWSLO_floor(iwca1))
     $       THEN
                ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
             ELSE
                ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
             ENDIF
             ws_bkp_frac_runff = 1.0
             if (.NOT. LWSWCA) THEN
               KFLO(IFLS2) = KFLO(IFLS2) - (runoff_south(is6bsn-6)
     $         -ws_down_s6)*ws_bkp_frac*ws_bkp_frac_runff
               flow_s2_bef_cap_lim = abs(KFLO(IFLS2))
               IF (KFLO(IFLS2) .LT. -0.000001) THEN
                  KFLO(IFLS2) = -amin1(abs(KFLO(IFLS2))
     $            ,rmax_capac_ws_bkp(is7bsn-6))
               ENDIF
               runoff_south(is6bsn-6) = flow_s2_bef_cap_lim
     $           - amin1(flow_s2_bef_cap_lim
     $           ,rmax_capac_ws_bkp(is7bsn-6))
     $           + (runoff_south(is6bsn-6)-ws_down_s6)*(1.0
     $           -ws_bkp_frac*ws_bkp_frac_runff)
             endif
            endif
           endif
           if (istaopt(is6bsn-6) .eq. 1) then
            runoff_s6bsn = runoff_south(is6bsn-6)
             call flow_to_stas(runoff_s6bsn+divers,ws_down_s6
     $   ,ws_env_to_wca2a,3,dmds236,s236so,volwtr,dellec,ws_from_lo_s6
     $   ,ifacteaa,kfloref,runoff_south,flow_from_eaa_lok_s8
     $   ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $   ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $   ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $   ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $   ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $   ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $   ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
           endif
c
```

```
            IF (ISTAOPT(IS6BSN-6) .EQ. 1) THEN
             UNTRTRF = HILLEXC
            ELSE
             UNTRTRF = KFLO(IFLS6)
            ENDIF
            untrtrf_south = untrtrf
c
c  determine water supply backpumping if no demands exist for wca or
c  lec
c
         IF (iwsbkpwsta .EQ. 0 .or. istaopt(is6bsn-6) .eq. 0) THEN
          if (LWSBPMP .and. KBFBPS2 .EQ. 0) THEN
            IF (LWSBKP_floor(is6bsn-6) .or. LWSLO_floor(iwca1))
     $      THEN
               ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is7bsn-6)
            ELSE
               ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is7bsn-6)
            ENDIF
            ws_bkp_frac_runff = 1.0
            if (.NOT. LWSWCA) THEN
              KFLO(IFLS2) = KFLO(IFLS2) - untrtrf*ws_bkp_frac
     $        *ws_bkp_frac_runff
              flow_s2_bef_cap_lim = abs(KFLO(IFLS2))
              IF (KFLO(IFLS2) .LT. -0.000001) THEN
                 KFLO(IFLS2) = -amin1(abs(KFLO(IFLS2))
     $           ,rmax_capac_ws_bkp(is7bsn-6))
              ENDIF
              if (istaopt(is6bsn-6) .eq. 0) then
                KFLO(IFLS6) = flow_s2_bef_cap_lim
     $          - amin1(flow_s2_bef_cap_lim
     $          ,rmax_capac_ws_bkp(is7bsn-6))
     $          + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $          *ws_bkp_frac_runff)
              endif
              untrtrf_south = flow_s2_bef_cap_lim
     $          - amin1(flow_s2_bef_cap_lim
     $          ,rmax_capac_ws_bkp(is7bsn-6))
     $          + (UNTRTRF - ws_down_s6)*(1.0-ws_bkp_frac
     $          *ws_bkp_frac_runff)
            endif
          endif
         ENDIF
        endif
c
      ELSE
        ws_down_sta2 = 0.0
        if (opt_for_hill_bypass .eq. 'FALSE') then
         if (istaopt(is6bsn-6) .eq. 1) then
          do irs = 1,nres_ws
          do js = 1,noutlets(irs)
            if (idshg_opt(ires_no(irs),js) .ne. 1
     $         .and. idshg_opt(ires_no(irs),js) .ne. 3) then
              call ws_from_res(irs,js,ires_no(irs),ieaa_bsn(irs,js)
     $         ,crmin,sumrain,jday,isem_flg,dmnd_sugh,dmnd_sugh_orig
     $         ,capac_semws,capac_sugh,stgref_asr_rec
     $         ,res_outflow_to_basin,stagelo_beg,pctmet
     $         ,sem_dmnd_orig,1,outflow_from_sta2_for_ws
```

```
     $         ,wsndes,wsndese,runoff_to_sta,volwtr
     $         ,frac_thru_s8_sta34,wsnde1,wsnde2,wsnde3,res_outflow_env
     $         ,rloc_excess_for_ws,tot_flw_to_c9r_frm_nlkblt
     $         ,ws_from_acmres_lwdd,issm_cutb_opt_bcyp
     $         ,pot_del_to_eaa_basin,alocnnr_potdel
     $         ,total_lec_dmnd_lok,iroute_to_wca3a_frm_holy
     $         ,iopt_env_eaar_thru_g372)
            endif
           enddo
           enddo
          endif
         endif
         if (iconv_use_inj(is6bsn-6) .eq. 0) then
          kfloref(is6bsn-6)=runoff_south(is6bsn-6)+divers
         else
          kfloref(is6bsn-6)=runoff_south_bef_divers(is6bsn-6)
         endif
         dmnd_hill = 0.0
         ratio_cap_desgn = 1.0
         LWSWCA = .FALSE.
         QS6DL = 0.0
         if (istaopt(is6bsn-6) .eq. 1) then
           runoff_s6bsn = runoff_south(is6bsn-6)
           tmpvar = 0.0
           call flow_to_stas (runoff_s6bsn+divers,0.0,tmpvar
     $ ,3,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
     $ ,kfloref,runoff_south,flow_from_eaa_lok_s8
     $ ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $ ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $ ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $ ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $ ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $ ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $ ,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
         endif
         flow_through_s6 = 0.0
c
         IF (ISTAOPT(IS6BSN-6) .EQ. 1) THEN
           UNTRTRF = HILLEXC
         ELSE
           UNTRTRF = KFLO(IFLS6)
         ENDIF
         untrtrf_south = untrtrf
       ENDIF
C
       if (wcaenv(idn_wca_indx_env(is6bsn-6)) .eq. 'TRUE ') then
         add_inflow = flow_through_s6
       else
         add_inflow = 0.0
       endif
       if ((istaopt(is6bsn-6) .eq. 0.or.runoff_south(is6bsn-6)+divers
     $ +add_inflow.le.dmnd_lec_s6).and.flow_to_wca_prior
     $ .ne. 'LEC' .and. ((((.not. LWSLO_floor(iwca1) .or. .not.
     $ LWSLECSTA(is6bsn-6)) .or. floor_grid_cell_opt.ne.'MARSH')
     $ .AND. rmin_stg_wca1 .le. rmnstg(iwca1,iconvey_canal_no_wca
     $ (iwca1)))).or. iopt_ws_lec_to_sta_wca_abv_flr .ne. 1) then
c      if ((runoff_south(is6bsn-6)+divers+add_inflow .le.
```

```
c     $    dmnd_lec_s6 .or. istaopt(is6bsn-6) .eq. 0) .and.
c     $    flow_to_wca_prior .ne. 'LEC') then
        if (istaopt(is6bsn-6) .eq. 0) then
           TFLOS6 = KFLO(IFLS6) + QS6DL
           KFLO(IFLS6) = AMIN1(TFLOS6,S6MAX)
           EXCDCAP = TFLOS6 - KFLO(IFLS6)
         if (opt_pipeline_ws(is6bsn-6) .eq. 'PIPEL  ') then
           ws_pipeline_ca1_hill = ws_pipeline_ca1_hill
      $          + qs6dl + amin1(kflo(ifls6),dmnd_lec_s6
      $          ,runoff_south(is6bsn-6))
           kflo(ifls6) = kflo(ifls6) - ws_pipeline_ca1_hill
         endif
         KFLO(IFLS2) = KFLO(IFLS2) + (QS6DL-EXCDCAP)
         kfloref(is6bsn-6) = kfloref(is6bsn-6) +(QS6DL-EXCDCAP)
         DELLEC = DELLEC + (QS6DL-EXCDCAP) * 1.9835
         WSS351 = WSS351 + (QS6DL-EXCDCAP)
         ws_to_lec_s351 = ws_to_lec_s351 + (QS6DL-EXCDCAP)
         ws_to_lec_s351_hill = QS6DL-EXCDCAP + ws_to_lec_s351_hill
         ws_from_lo_s6 = qs6dl - excdcap
         flow_s6 = kflo(ifls6) + ws_pipeline_ca1_hill
        else
         KFLO(IFLS2) = KFLO(IFLS2) + QS6DL
         kfloref(is6bsn-6) = kfloref(is6bsn-6) + QS6DL
         DELLEC = DELLEC + (QS6DL) * 1.9835
         ws_to_lec_s351 = ws_to_lec_s351 + QS6DL
         ws_to_lec_s351_hill = QS6DL + ws_to_lec_s351_hill
         WSS351 = WSS351 + QS6DL
         if (opt_for_hill_bypass .eq. 'TRUE ') then
           ws_from_lo_s6 = qs6dl+untrtrf_south
         else
           if (opt_for_hill_bypass_runoff .eq. 'TRUE ') then
             ws_from_lo_s6 = untrtrf_south
           else
             ws_from_lo_s6 = 0.0
           endif
           flow_s6 = ws_from_lo_s6
         endif
         if (opt_pipeline_ws(is6bsn-6) .ne. 'PIPEL  ') then
           qu(ica1) = qu(ica1) + (ws_from_lo_s6)* 86400.
         else
           ws_pipeline_ca1_hill = ws_pipeline_ca1_hill
      $        + ws_from_lo_s6
         endif
         flow_s6 = ws_from_lo_s6
         if (iconvey_canal_no(is6bsn-6) .ne. 0) then
          addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))
      $   = addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))
      $     + ws_from_lo_s6
         endif
        endif
       else
         if (opt_for_hill_bypass .eq. 'TRUE ' .or.
      $     opt_for_hill_bypass_runoff .eq. 'TRUE ') then
           flow_s6 = ws_from_lo_s6
         else
           flow_s6 = 0.0
           ws_from_lo_s6 = 0.0
```

```
        endif
        if (iconvey_canal_no(is6bsn-6) .ne. 0) then
         addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))
     $   = addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))
     $     + flow_s6
        endif
      endif
      avvol_upstream = -901.
      cname = 'hillsb'
      str_name = 's351h '
      total_ws_to_s6bsn = dmnd_hill + wsdel_to_sta(indxsta(is6bsn-6))
     $ + wsdel_to_sta(indx_sta2w_frm_lok_ws)
      if (istaopt(is6bsn-6) .eq. 1) then
         total_outflow_for_s6 = flow_s6 + kflo(ifls6)
      else
         total_outflow_for_s6 = kflo(ifls6)
      endif
      if (dmnd_hill+add_red_in_conv .ge. fllks6) fllks6 =
     $          total_outflow_for_s6  + total_ws_to_s6bsn
     $          + add_red_in_conv
      if (lp(23)) then
      write(62,235) ifyr+iyear-1,month,iday,str_name,stagelo
     $  ,s351_tw,s351cap-total_flow_s2,avvol_upstream
     $  ,total_ws_to_s6bsn+qs6dl,dmnd_lec_s6+wsuply(indxsta(is6bsn-6))
     $  +wsuply(indx_sta2w_frm_lok_ws)
      write(62,235) ifyr+iyear-1,month,iday,cname,s351_tw,s6_hw
     $  ,fllks6,avvol_upstream,total_outflow_for_s6+total_ws_to_s6bsn
     $  +add_red_in_conv,dmnd_lec_s6+wsuply(indxsta(is6bsn-6))
      write(62,235) ifyr+iyear-1,month,iday,flnm(ifls6),s6_hw
     $  ,s6_tailw,s6max,avvol_upstream,flow_s6
     $  ,dmnd_lec_s6
      endif
C
C WPB CANAL TO WCA-1 THROUGH S-352 AND S5A
C
C RLKDLWS = TOTAL DEMAND IN CA1 AND LEC
C QS6DL = PORTION OF RLKDLWS TO BE SATISFIED THROUGH HILLSBORO CANAL
C QS5ADL = PORTION OF RLKDLWS TO BE SATISFIED THROUGH WEST PALM BEACH CANAL
C
C
      IF (LWSBPMP) THEN
         IF (LWSBKP_floor(is5absn-6) .or. LWSLO_floor(iwca1)) THEN
            ws_bkp_frac = frac_ws_bkp_bel_wcaflr(is5absn-6)
         ELSE
            ws_bkp_frac = frac_ws_bkp_abv_wcaflr(is5absn-6)
         ENDIF
         ws_bkp_frac_runff = 1.0
      ENDIF
c
      IF (nint(RLKDLWS_SA1/86400.) .GT. 0) THEN
        IF (opt_for_hill_bypass .eq. 'TRUE ') then
         QDELHG5 = RLKDLWS_SA1/86400. - (QS6DL-EXCDCAP)
          if (opt_for_hill_bypass_runoff .eq. 'TRUE ') then
             QDELHG5 = QDELHG5 - untrtrf_south
          endif
        else
          QDELHG5 = RLKDLWS_SA1/86400. - (WSNDES - WSNDES_REV)
```

```
     $               - (WSNDESE - WSNDESE_REV)
        ENDIF
        QDELHG5 = AMAX1(QDELHG5,0.0)
      ELSE
         QDELHG5 = 0.0
      ENDIF
      dmnd_lec_s5a = RLKDLWS_SA1_WO_UNTRTRFS5A - (QS6DL-EXCDCAP)
      BFLOW_S352  = 0.0
      bflo_via_l8 = 0.0
C
      iconvey_c = iconvey_canal_no(is5absn-6)
      rem_capac_s352 = amax1(S352MX-KFLO(IFLHG5),0.0)
      if (is5absn-6 .ge. 2) then
         do isindx = is5absn-6,2,-1
          if (ieaa_res_asr(is5absn-6) .eq. ieaa_res_asr(isindx-1)) then
            inflow_indx = inflow_indx + 1
          endif
         enddo
      endif
      runoff_south_bef_divers(is5absn-6) = runoff_south(is5absn-6)
c
c  determine if criteria is met for environmental deliv
c   fom LOK to WCA-1 thru WPB Canal in EAA
c
       ntarg = 0
       if (lrunoff_to_res_when_above_targ .eq. 'FALSE') then
          do itarg = 1,no_targ_loc(is5absn-6)
            if (LWSLO(itarg_area(itarg,is5absn-6))) then
               ntarg = ntarg + 1
            endif
          enddo
       else
          do itarg = 1,no_targ_loc(is5absn-6)
             if (avg_sim_wca_stg(itarg_area(itarg,is5absn-6))
     $         .lt. stage_import_wca(itarg_area(itarg,is5absn-6))
     $         + offset(itarg_area(itarg,is5absn-6),1)) then
                ntarg = ntarg + 1
             endif
          enddo
       endif
c
       if  (wcaenv(idn_wca_indx_env(is5absn-6)) .eq. 'FALSE') then
c
c route excess eaa runoff to asr reservoir
c
        if (iasr_opt_eaa(is5absn-6) .eq. 1 .or. ires_opt_eaa
     $      (is5absn-6) .eq. 1) then
           runoff_to_inject = amax1(runoff_south(is5absn-6)
     $         - dmnd_lec_s5a - wsuply(indxsta(is5absn-6)),0.0)
           flow_to_eaa_res_rnff(is5absn-6) = amin1(runoff_to_inject
     $      *frac_runoff_asr(is5absn-6),volwtr(ieaa_res_asr(is5absn-6))
     $      ,weircff(ieaa_res_asr(is5absn-6),inflow_indx,1))
           rem_weir_capac(is5absn-6) =weircff(ieaa_res_asr(is5absn-6)
     $          ,inflow_indx,1) - flow_to_eaa_res_rnff(is5absn-6)
           volwtr(ieaa_res_asr(is5absn-6))
     $      = volwtr(ieaa_res_asr(is5absn-6))-flow_to_eaa_res_rnff
     $      (is5absn-6)
```

```
            runoff_south(is5absn-6) = runoff_south(is5absn-6)
     $         - flow_to_eaa_res_rnff(is5absn-6)
            q = runoff_south(is5absn-6)
            IF( Q .GT. S5AMAX) THEN
               KFLO(IFLS5A) = S5AMAX * PCTWMA(IS5ABSN-6)
               Q = Q - S5AMAX * PCTWMA(IS5ABSN-6)
               IAGFLO(IAG(IS5ABSN),1) = NINT(Q) + IAGFLO(IAG
     $                (IS5ABSN),1)
            ELSE
               KFLO(IFLS5A) = Q
            ENDIF
            eaa_runff_s5a_to_sta = KFLO(IFLS5A)
            KFLO(IFLS5A5) = KFLO(IFLS5A5) + KFLO(IFLS5A)
            s5a2_so = s5a2_so + kflo(ifls5a)
         endif
        else
c
c route excess eaa runoff to asr reservoir
c
           if (iasr_opt_eaa(is5absn-6) .eq. 1) then
c            if (.not. LWSLO(iwca1)) then
             if (ntarg .eq. 0) then
               runoff_to_inject = amax1(runoff_south(is5absn-6)
     $            - dmnd_lec_s5a - wsuply(indxsta(is5absn-6)),0.0)
             else
               rmax_flow_thru = amin1(rem_capac_s352,FLLKWPB
     $                - KFLO(IFLHG5)-wsuply(indx_sta1w_frm_lok_ws)
     $                - wsuply(indx_sta1e_frm_lok_ws)
     $                - wsuply(indx_sta1ee_frm_lok_ws),s5amax)
               rmax_flow_thru = amax1(rmax_flow_thru,0.0)
               rmax_flow_bypass_res = amax1(rmax_flow_thru,dmnd_lec_s5a)
               runoff_to_inject = runoff_south(is5absn-6)
     $                   - rmax_flow_bypass_res
               runoff_to_inject = amax1(runoff_to_inject,0.0)
               if (divers_excess_to_res .eq. 'FALSE') then
                 runoff_to_inject = 0.0
               endif
             endif
             flow_to_eaa_res_rnff(is5absn-6) = amin1(runoff_to_inject
     $       *frac_runoff_asr(is5absn-6),volwtr(ieaa_res_asr(is5absn-6))
     $       ,weircff(ieaa_res_asr(is5absn-6),inflow_indx,1))
            rem_weir_capac(is5absn-6) =weircff(ieaa_res_asr(is5absn-6)
     $          ,inflow_indx,1) - flow_to_eaa_res_rnff(is5absn-6)
             volwtr(ieaa_res_asr(is5absn-6))
     $       = volwtr(ieaa_res_asr(is5absn-6))-flow_to_eaa_res_rnff
     $       (is5absn-6)
             runoff_south(is5absn-6) = runoff_south(is5absn-6)
     $         - flow_to_eaa_res_rnff(is5absn-6)
            q = runoff_south(is5absn-6)
            IF( Q .GT. S5AMAX) THEN
               KFLO(IFLS5A) = S5AMAX * PCTWMA(IS5ABSN-6)
               Q = Q - S5AMAX * PCTWMA(IS5ABSN-6)
               IAGFLO(IAG(IS5ABSN),1) = NINT(Q) + IAGFLO(IAG
     $                (IS5ABSN),1)
            ELSE
               KFLO(IFLS5A) = Q
            ENDIF
```

```
            eaa_runff_s5a_to_sta = KFLO(IFLS5A)
            KFLO(IFLS5A5) = KFLO(IFLS5A5) + KFLO(IFLS5A)
            s5a2_so = s5a2_so + kflo(ifls5a)
           endif
        endif
        IF (LWSBKP_floor(is5absn-6) .or. LWSLO_floor(iwca1))
     $ THEN
          frac_bkflw_wpb_via_l8 = frac_bkflw_wpb_via_l8_bel_flr
        ELSE
          frac_bkflw_wpb_via_l8 = frac_bkflw_wpb_via_l8_abv_flr
        ENDIF
        IF( runoff_south(is5absn-6).GT. dmnd_lec_s5a) THEN
          IF(LWSBPMP.AND. STAGELO .LT. 11.5 .and. ntarg .eq. 0) THEN
C WS BP IS DESIRED AND POSSIBLE SINCE S3 IS NOT YET UTILIZED AT THIS POINT;
C REDISTRIBUTE FLOWS BETWEEN S8 AND S3 BY CONSIDERING RLKDLWS
C PARTITION S8 FLOW TO SOUTH (I.E. WHAT GOES TO MEET ALL WCA AND LEC DEMAND)
C AND TO NORTH (I.E. WHAT GOES AS WATER SUPPLY BACKPUMPING)
              BFLOW_S352 = amin1((runoff_south(is5absn-6)-dmnd_lec_s5a)
     $        *ws_bkp_frac*ws_bkp_frac_runff,rmax_capac_ws_bkp
     $        (is5absn-6))
              KFLO(IFLHG5)=KFLO(IFLHG5)-BFLOW_S352
              KFLO(IFLS5A) = kflo(ifls5a) - BFLOW_S352
              if (chdep(il8) .lt. 14.0 .and. .not. opt_l8_prop) then
                bflo_via_l8 = frac_bkflw_wpb_via_l8 *amax1(runoff_south
     $          (is5absn-6)- dmnd_lec_s5a,0.0)
                bflo_via_l8 = amin1(bflo_via_l8,bflo_cap_l8)
              endif
              runoff_south(is5absn-6) = runoff_south(is5absn-6)
     $        - (bflo_via_l8+BFLOW_S352)
              QDELHG5 = 0.0
              KFLO(IFLS5A5) = KFLO(IFLS5A5) - BFLOW_S352 - bflo_via_l8
              s5a2_so = s5a2_so - BFLOW_S352 - bflo_via_l8
              KFLO(IFLL8) = KFLO(IFLL8) - bflo_via_l8
              s5a3_no = s5a3_no + bflo_via_l8
          ELSE
              IF (LWSBPMP .and. ntarg .eq. 0) THEN
               if (chdep(il8) .lt. 14.0 .and. .not. opt_l8_prop) then
                 bflo_via_l8 = (frac_bkflw_wpb_via_l8+ws_bkp_frac)
     $           *amax1(runoff_south(is5absn-6)- dmnd_lec_s5a,0.0)
                 bflo_via_l8 = amin1(bflo_via_l8,bflo_cap_l8)
                 runoff_south(is5absn-6) = runoff_south(is5absn-6)
     $           - bflo_via_l8
                 KFLO(IFLS5A5) = KFLO(IFLS5A5) - bflo_via_l8
                 s5a2_so = s5a2_so - bflo_via_l8
                 KFLO(IFLL8) = KFLO(IFLL8) - bflo_via_l8
                 s5a3_no = s5a3_no + bflo_via_l8
               endif
              ENDIF
C S8 FLOW CANNOT MEET TOTAL WCA AND LEC REQUIREMENT; NO WS BACKPUMPING POSSIBLE;
C JUST REDUCE WS NEED BASED ON S8 CONTRIBUTION
              QDELHG5 = QDELHG5 - amin1(QDELHG5,runoff_south(is5absn-6))
          ENDIF
          eaa_runff_s5a_to_sta = eaa_runff_s5a_to_sta - bflo_via_l8
     $   - BFLOW_S352
        ELSE
          QDELHG5 = QDELHG5 - amin1(QDELHG5,runoff_south(is5absn-6))
          bflo_via_l8 = 0.0
```

```
        ENDIF
c       print *,frac_bkflw_wpb_via_l8,runoff_south(is5absn-6)
c     $ ,dmnd_lec_s5a,STAGELO,ntarg,ws_bkp_frac,ws_bkp_frac_runff
c     $ ,LWSBPMP,BFLOW_S352,bflo_via_l8
c
        ntarg = 0
        do itarg = 1,no_targ_loc(is5absn-6)
          if (LWSLO(itarg_area(itarg,is5absn-6))) then
             ntarg = ntarg + 1
          endif
        enddo
c
       if (iconv_use_inj(is5absn-6) .eq. 0) then
        kfloref(is5absn-6) = amax1(kflo(iflhg5),kflo(ifls5a))
      $ + BFLOW_S352
       else
        kfloref(is5absn-6) = amax1(kflo(iflhg5)+BFLOW_S352,
      $             runoff_south_bef_divers(is5absn-6))
       endif
       add_conv_red_wpb =  0.0
       do irs = 1,nres_ws
         if (iconv_use_opt_ws(irs) .eq. 1) then
           do js = 1,noutlets(irs)
             if (ieaa_bsn(irs,js) .eq. is5absn-6) then
               add_conv_red_wpb = add_conv_red_wpb
      $                + res_outflow_to_basin(irs,js)
             endif
           enddo
         endif
       enddo
       kfloref(is5absn-6) = kfloref(is5absn-6) + add_conv_red_wpb
c
c  water supply from LOK to STA1W
c
       if (istaopt(is5absn-6) .eq. 1 .and. BFLOW_S352 .lt. 0.00001) then
         wsdel_to_sta(indx_sta1w_frm_lok_ws) = amin1(wsuply
      $     (indx_sta1w_frm_lok_ws),amax1(fllkwpb-kfloref(is5absn-6),0.0)
      $     ,s352mx-kflo(iflhg5))
         kfloref(is5absn-6) = kfloref(is5absn-6)
      $           + wsdel_to_sta(indx_sta1w_frm_lok_ws)
         kflo(iflhg5) = kflo(iflhg5) + wsdel_to_sta
      $     (indx_sta1w_frm_lok_ws)
         wss352 = wss352 + wsdel_to_sta(indx_sta1w_frm_lok_ws)
         wsdel_to_sta(indx_sta1e_frm_lok_ws) = amin1(wsuply
      $      (indx_sta1e_frm_lok_ws),amax1(fllkwpb-
      $      kfloref(is5absn-6),0.0),s352mx-kflo(iflhg5))
         kflo(iflhg5) = kflo(iflhg5) + wsdel_to_sta
      $      (indx_sta1e_frm_lok_ws)
         wsdel_to_sta(indx_sta1ee_frm_lok_ws) = amin1(wsuply
      $      (indx_sta1ee_frm_lok_ws),amax1(fllkwpb-
      $      kfloref(is5absn-6),0.0),s352mx-kflo(iflhg5))
         kflo(iflhg5) = kflo(iflhg5) + wsdel_to_sta
      $      (indx_sta1ee_frm_lok_ws)
         call match (name_res_spec_code(6),1,6,resname,ntotres,6
      $           ,ista1e_res)
         if (ista1e_res .ne. 0)
      $  res_struc_inflw(ista1e_res) = res_struc_inflw(ista1e_res)
```

```
     $    + wsdel_to_sta(indx_sta1e_frm_lok_ws)*1.9835
        wss352 = wss352 + wsdel_to_sta(indx_sta1e_frm_lok_ws)
     $    + wsdel_to_sta(indx_sta1ee_frm_lok_ws)
        kfloref(is5absn-6) = kfloref(is5absn-6)
     $    + wsdel_to_sta(indx_sta1e_frm_lok_ws)
     $    + wsdel_to_sta(indx_sta1ee_frm_lok_ws)
      endif
      if (wcaenv(idn_wca_indx_env(is5absn-6)) .eq. 'TRUE ') then
        oper_capac = cnlcap(iwpb) - (1.0-pct_des(iwpb,iseas))
     $                     *cnlcap(iwpb)/rcpfacteaa(is5absn-6)
        if (.NOT. DRYSEAS .and. nint(runoff_south(is5absn-6))
     $       .gt. 0) then
          rmax_canal_capac = amin1(fllkwpb,oper_capac)
     $       - kfloref(is5absn-6)
        else
          rmax_canal_capac = amin1(fllkwpb - kfloref(is5absn-6)
     $           ,oper_capac)
        endif
        rmax_canal_capac = amax1(rmax_canal_capac,0.0)
        if (nasr_to_lok(1) .eq. 0) then
          if (stagelo .lt. stgref_asr_rec .and. (climate_index(month)
     $      .lt. 1.5 .or. ((itrib_hydro_cond_opt .eq. 1 .and.
     $      trib_rf_et .lt. -1.0 .and.
     $      s65e_runff_wkly .lt. 1500.) .or.
     $      itrib_hydro_cond_opt .eq. 0)) .and. DRYSEAS) then
           frac_env_deliv = 0.00
          else
            frac_env_deliv = 1.00
          endif
        else
          frac_env_deliv = 1.00
        endif
        frac_env_deliv = amin1(frac_env_deliv,frac_env_met)
c       if (LWSLO(iwca1)) then
        if (ntarg .gt. 0) then
          QS5ADL = amin1(rem_capac_s352,rmax_canal_capac)
     $             * frac_rem_capac(is5absn-6) * frac_env_deliv
     $           *frac_del_frm_wca(iwca1)
          flow_to_import = flow_to_import + qs5adl
          flow_to_import_wpb = qs5adl
        else
          avail_canal_cap = amax1(FLLKWPB-kfloref(is5absn-6),0.0)
          QS5ADL = amin1(rem_capac_s352,avail_canal_cap
     $               ,QDELHG5)*frac_del_frm_wca(iwca1)
        endif
      else
        avail_canal_cap = amax1(FLLKWPB-kfloref(is5absn-6),0.0)
        QS5ADL = AMIN1(rem_capac_s352,QDELHG5
     $             ,avail_canal_cap)*frac_del_frm_wca(iwca1)
      endif
c     print *,ntarg,QS5ADL,rem_capac_s352,rmax_canal_capac
c    $,'  S352EV2  ',QDELHG5,avail_canal_cap,kfloref
c    $,fllkwpb
C
C NOTE: KFLO(IFLHG5) >= 0 ALWAYS
C
      TFLOS5A = KFLO(IFLS5A) + QS5ADL
```

```
      KFLO(IFLS5A) = AMIN1(TFLOS5A,S5AMAX)
      EXCDCAP = TFLOS5A - KFLO(IFLS5A)
      KFLO(IFLS5A5) = KFLO(IFLS5A5) + QS5ADL - EXCDCAP
      s5a2_so = s5a2_so + qs5adl - excdcap
      if (opt_pipeline_ws(is5absn-6) .eq. 'PIPEL  ') then
        ws_pipeline_ca1_wpb = ws_pipeline_ca1_wpb
     $         + amin1(QS5ADL ,QDELHG5)
        KFLO(IFLS5A) = KFLO(IFLS5A) - ws_pipeline_ca1_wpb
        KFLO(IFLS5A5) = KFLO(IFLS5A5) - ws_pipeline_ca1_wpb
        s5a2_so = s5a2_so - ws_pipeline_ca1_wpb
      endif
      KFLO(IFLHG5) = KFLO(IFLHG5) + QS5ADL - EXCDCAP
      kfloref(is5absn-6) = kfloref(is5absn-6) + QS5ADL - EXCDCAP
      flow_to_lec = qs5adl - excdcap
      avail_flow_to_lec = flow_to_lec
c
      if (istaopt(is5absn-6) .eq. 1) then
       if (wcaenv(iwca1) .eq. 'TRUE ') then
         if (ntarg .gt. 0) then
           avail_flow_to_lec = frac_lec(is5absn-6)*(kflo(ifls5a)
     $                          + ws_pipeline_ca1_wpb)
         else
           avail_flow_to_lec = kflo(ifls5a)+ws_pipeline_ca1_wpb
         endif
       else
          avail_flow_to_lec = QS5ADL - EXCDCAP
       endif
        flow_to_lec = amin1(avail_flow_to_lec,QDELHG5)
        if ((((.not. LWSLO_floor(iwca1) .or. .not.
     $  LWSLECSTA(is5absn-6)) .or. floor_grid_cell_opt.ne.'MARSH')
     $  .AND. rmin_stg_wca1 .le. rmnstg(iwca1,iconvey_canal_no_wca
     $  (iwca1))).or. iopt_ws_lec_to_sta_wca_abv_flr .ne. 1) then
         if (opt_pipeline_ws(is5absn-6) .ne. 'PIPEL  ') then
          rchg(nodl101) = rchg(nodl101) - flow_to_lec
     $            *86400./gdar
          qu(ica1) = qu(ica1) + flow_to_lec*86400.
         endif
         if (iconvey_canal_no(is5absn-6) .ne. 0) then
           addflw(idn_wca_indx(is5absn-6),iconvey_canal_no(is5absn-6))
     $     = addflw(idn_wca_indx(is5absn-6),iconvey_canal_no(is5absn-6))
     $     + flow_to_lec
         endif
         s5a_to_sta = kflo(ifls5a)+ws_pipeline_ca1_wpb - flow_to_lec
        else
         s5a_to_sta = kflo(ifls5a)+ws_pipeline_ca1_wpb
         flow_to_lec = 0.0
        endif
      endif
c
      if (ssm_env .eq. 'TRUE ') then
         DELLEC = DELLEC + (QS5ADL - EXCDCAP) * 1.9835
      else
         avail_flow_to_lec = amin1(avail_flow_to_lec,QS5ADL - EXCDCAP)
         DELLEC = DELLEC + AMIN1(avail_flow_to_lec,QDELHG5)*1.9835
      endif
      ws_to_lec_s352 = AMIN1(QS5ADL - EXCDCAP,QDELHG5)
      if (ntarg .gt. 0) then
```

```
c
        flow_to_import = flow_to_import - ws_to_lec_s352
        flow_to_import_wpb = flow_to_import_wpb - ws_to_lec_s352
c
      endif
c
      ws_from_lo_s5a = flow_to_lec
      WSS352 = QS5ADL - EXCDCAP + WSS352
      if (iconvey_canal_no(is6bsn-6) .ne. 0) then
       if (runoff_south(is6bsn-6) +divers .lt. dmnd_lec_s6) then
        lec_dmnd_on_lok(iwca1,iconvey_canal_no(is6bsn-6)) = 1
       else
        lec_dmnd_on_lok(iwca1,iconvey_canal_no(is6bsn-6)) = 0
       endif
       if (runoff_south(is5absn-6) .lt. dmnd_lec_s5a) then
        lec_dmnd_on_lok(iwca1,iconvey_canal_no(is5absn-6)) = 1
       else
        lec_dmnd_on_lok(iwca1,iconvey_canal_no(is5absn-6)) = 0
       endif
      endif
c
      avvol_upstream = -901.
      cname = 'wpbcnl'
      total_ws_to_s5a = kflo(iflhg5)
      if (lp(23)) then
      write(62,235) ifyr+iyear-1,month,iday,flnm(iflhg5),stagelo
     $  ,s352_tw,s352cap,avvol_upstream,total_ws_to_s5a,qdelhg5
     $  +wsuply(indxsta(is5absn-6))
      fllkwpb_ws = fllkwpb
      if (total_ws_to_s5a .gt. fllkwpb) fllkwpb_ws = total_ws_to_s5a
      write(62,235) ifyr+iyear-1,month,iday,cname,s352_tw,s5a_hw
     $ ,fllkwpb_ws,avvol_upstream,amax1(kflo(ifls5a)
     $ +ws_pipeline_ca1_wpb,total_ws_to_s5a)
     $ ,qdelhg5+wsuply(indx_sta1w_frm_lok_ws)
     $ +wsuply(indx_sta1e_frm_lok_ws)+wsuply(indx_sta1ee_frm_lok_ws)
      write(62,235) ifyr+iyear-1,month,iday,flnm(ifls5a),s5a_hw
     $ ,s5a_tailw,s5amax,avvol_upstream,kflo(ifls5a)
     $  +ws_pipeline_ca1_wpb,qdelhg5
      endif
        if (opt_pipeline_ws(is5absn-6) .eq. 'PIPEL  ') then
           kflo(ifls5a5) = kflo(ifls5a5) - untreated_runoff_thru_s5a
           call match (name_res_spec_code(8),1,6,resname,ntotres,6
     $           ,ista1w_res)
           rchg(nodepw(ista1w_res,1,1)) = rchg(nodepw(ista1w_res,1,1))
     $      - amin1(out_to_wca1_frm_sta1w,wslec_sa1/86400.)*86400./(gdar
     $      * (1.0 - carfact(nodepw(ista1w_res,1,1))
     $      - total_small_afact(nodepw(ista1w_res,1,1))))
           rem_wslec_sa1 = wslec_sa1/86400. -amin1(out_to_wca1_frm_sta1w
     $                  ,wslec_sa1/86400.)
           call match (name_res_spec_code(6),1,6,resname,ntotres,6
     $           ,ista1e_res)
           rchg(nodepw(ista1e_res,1,1)) = rchg(nodepw(ista1e_res,1,1))
     $      - amin1(out_to_wca1_frm_sta1e,rem_wslec_sa1)*86400./(gdar
     $      * (1.0 - carfact(nodepw(ista1e_res,1,1))
     $      - total_small_afact(nodepw(ista1e_res,1,1))))
        endif
c
```

```
c     route runoff from 298 districts to sta34 and sta2
c     if (nmarea .gt. 0) then
c       tmpvar = 0.0
c       call flow_to_stas(runoff_298_districts,0.0,tmpvar
c    $ ,4,dmds236,s236so,volwtr,dellec,ws_from_lo_s6,ifacteaa
c    $ ,kfloref,runoff_south,flow_from_eaa_lok_s8
c    $ ,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
c    $ ,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
c    $ ,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
c    $ ,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
c    $ ,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
c    $ ,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
c    $ ,rmin_stg_wca1)
c     endif
c
c     G-155,G-89,G-88 to STA-5 and runoff from sugar ranch
c         to STA-6
c
      tmpvar = 0.0
      call flow_to_stas(-901.0,0.0,tmpvar,5,dmds236,s236so,volwtr
     $,dellec, ws_from_lo_s6,ifacteaa,kfloref,runoff_south
     $,flow_from_eaa_lok_s8,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $,ws_to_sta34_via_nnrc,add_conv_red_miami,ws_to_sta34_frm_lok
     $,water_supply_to_holyl,frac_env_met,dmnd_lec_sa2_s7
     $,dmnd_lec_sa3_s150,pctmet,climate_index,iday_lok_in_ssm
     $,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a,rmin_stg_wca3a
     $,rmin_stg_wca1,dmnd_nnr_aft_resasr,res_outflow_to_basin)
c
c     route makeup water to stas
c
      tmpvar = 0.0
      call flow_to_stas(-901.0,0.0,tmpvar,6,dmds236,s236so,volwtr
     $,dellec,ws_from_lo_s6,ifacteaa,kfloref,runoff_south
     $,flow_from_eaa_lok_s8,flow_from_eaa_lok_s7,ws_frm_lok_to_semcyp
     $,avail_cap_for_sta34_nnr,avail_cap_for_sta34_miami
     $,ws_to_sta34_via_nnrc,add_conv_red_miami
     $,ws_to_sta34_frm_lok,water_supply_to_holyl,frac_env_met
     $,dmnd_lec_sa2_s7,dmnd_lec_sa3_s150,pctmet,climate_index
     $,iday_lok_in_ssm,bmp_ssm_opt,LWSLECSTA,rmin_stg_wca2a
     $,rmin_stg_wca3a,rmin_stg_wca1,dmnd_nnr_aft_resasr
     $,res_outflow_to_basin)
c
c  determine portion of Taylor Creek/Nubbin Slough outflow
c     into Lake Okeechobee when appropriate
c
      if (itcns_res_opt .eq. 1) then
       if (day1) hrest0 = 0.0
       call taycrnbs_res(tcnsrnff(jday),tayck_nubsl_res_area
     $    ,0.0,10.0,hrest0,norstorrf(jday),norstorpet(jday)
     $    ,2500.,1000., actual_et,actual_seepage,res2lok
     $    ,infltores, hres,stagelo,stgref_nstor_to_lok
     $    ,month,ifyr, iyear, iday)
       hrest0 = hres
      else
        res2lok = 0.0
      endif
```

```
      tcrnbsres_to_lok = res2lok
c
c  route environmental water supply to Rotenberger Tract via STA-5
c
       if (opt_ws_to_roten_frm_lok .eq. 'YES') then
         IF (ic139_rnff_fix_split_opt .eq. 0) then
          TOTFLOW = amax1(KFLO(IG155),0.0) + KFLO(IG89) + KFLO(IG88)
         ELSE
          TOTFLOW = (amax1(KFLO(IG155),0.0)+KFLO(IG89)+KFLO(IG88))
     $      * frac_c139basin_sta5
         ENDIF
          call bpts(iday,month,10.0,8,1,1,roten_sched)
          IF (POND(node_roten1) .GT. DPTHGW) THEN
            stage_roten1 = POND(node_roten1) - DPTHGW +ELLS(node_roten1)
          ELSE
            stage_roten1 = (POND(node_roten1)-DPTHGW)/S(node_roten1)
     $             * ells(node_roten1)
          ENDIF
          DPTHGW = (ELLS(node_roten2) - H(node_roten2))
     $                * S(node_roten2)
          IF (POND(node_roten2) .GT. DPTHGW) THEN
            stage_roten2 = POND(node_roten2) - DPTHGW +ELLS(node_roten2)
          ELSE
            stage_roten2 = (POND(node_roten2)-DPTHGW)/S(node_roten2)
     $             * ells(node_roten2)
          ENDIF
          roten_stage = (stage_roten1 + stage_roten2)/2.0
          if (rotenenv .eq. 'TRUE ') then
            roten_import_stage = stage_import_wca(iroten)
            roten_stage = avg_sim_wca_stg(iroten)
          else
            roten_import_stage = roten_sched
          endif
          rem_capac_ws = amax1(2510. - totflow,0.0)
          if (roten_stage .lt. roten_import_stage-offset(iroten,1)
     $              - 0.25) then
             ws_to_roten_frm_lok = rem_capac_ws
          elseif (roten_stage .lt. roten_import_stage
     $              -offset(iroten,1)) then
             ws_to_roten_frm_lok = 4.0 * rem_capac_ws
     $              * (roten_import_stage-offset(iroten,1)
     $              - roten_stage)
          else
             ws_to_roten_frm_lok = 0.0
          endif
          ws_to_roten_frm_lok = amin1(ws_to_roten_frm_lok
     $          ,volwtr(indxsta(6)))
          qinflw(indxsta(6),1) = qinflw(indxsta(6),1)
     $                + ws_to_roten_frm_lok
          volwtr(indxsta(6)) = volwtr(indxsta(6)) - ws_to_roten_frm_lok
        else
          ws_to_roten_frm_lok = 0.0
        endif
c
        dellec = dellec + ws_to_roten_frm_lok*1.9835
     $   + wsdel_to_sta(indxsta(is8bsn-6))*1.9835
     $   + wsdel_to_sta(indxsta(is7bsn-6))*1.9835
```

```
     $   + wsdel_to_sta(indxsta(is6bsn-6))*1.9835
     $   + wsdel_to_sta(indx_sta1w_frm_lok_ws)*1.9835
     $   + dmdsemind(jday) + wsdel_to_sta(indx_sta2w_frm_lok_ws)
     $   * 1.9835
c
c   sum lec demand on LOK
c
        total_lec_dmnd_lok = dmnd_lec_sa3+dmnd_lec_sa2+dmnd_lec_s6
     $     + dmnd_lec_s5a
C
C UPDATE STORAGE IN LAKE OKEECHOBEE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      if (lok_split_option .eq. 0) then
       S1 = SPREV-(KFLO(IFLS3)+KFLO(IFLS2)+KFLO(IFLHG5)
     &   + KFLO(IFLAGQ)+KFLO(iflc10a))*1.9835-(eqs308 + eqs77)
     &   + s65ernff(jday) + s4bsn_to_lok - DMDS4 - DMDS236
     &   - dmdsemind(jday) + tcnsrnff(jday) + tcrnbsres_to_lok*1.9835
     &   - ws_to_roten_frm_lok*1.9835 - fplresws(jday) +
     &   chgrnfistapoga(jday) - chgdmdistapoga(jday) + chgrnfnls(jday) -
     &   chgdmdnls(jday) + chgrnfnels(jday) - chgdmdnels(jday)
     &   + uisrnff(jday)
      else
       stagelo_res_beg = stagelo_res
       if (stagelo .gt. stgref .or. stagelo_res .lt. 2.0) then
         vol_flow_litzone = -(KFLO(IFLS3)+KFLO(IFLS2)
     $    + KFLO(IFLHG5)+KFLO(IFLAGQ)+amax1(KFLO(iflc10a),0.0))*1.9835
     $    -(amax1(eqs308,0.0)+eqs77)+s65ernff(jday)+s4bsn_to_lok-DMDS4
     $    - dmdsemind(jday) + tcnsrnff(jday) + tcrnbsres_to_lok*1.9835
     $    - ws_to_roten_frm_lok*1.9835 - fplresws(jday) - DMDS236
     $    + eallokres1 + eallokres2 + c43dmd_met_by_lokres
     $    + c44dmd_met_by_lokres + chgrnfistapoga(jday)
     $    - chgdmdistapoga(jday) + chgrnfnls(jday) -
     $   chgdmdnls(jday) - chgdmdnels(jday) + uisrnff(jday)
         vol_flow_res =  - amin1(eqs308,0.0) - amin1(KFLO(iflc10a),0.0)
     $                     * 1.9835 - eallokres1 - eallokres2
     $                     - c43dmd_met_by_lokres - c44dmd_met_by_lokres
     $                     + chgrnfnels(jday)
         s354_res = 0.0
         s351_res = 0.0
         s352_res = 0.0
         dmnds4_res = 0.0
         dmnds236_res = 0.0
         dmndsemind_res = 0.0
         rlokreswsroten = 0.0
         flpresws_res = 0.0
         r298dws_res = 0.0
         l8ws_res = 0.0
       else
         vol_flow_litzone = -eqs77-amax1(eqs308,0.0)
     $    + s65ernff(jday)-amin1(KFLO(IFLS3),0.0)*1.9835
     $    - amin1(KFLO(IFLS2),0.0)*1.9835 + tcrnbsres_to_lok*1.9835
     $    + tcnsrnff(jday) + s4bsn_to_lok - amin1(KFLO(IFLAGQ),0.0)
     $    * 1.9835 + eallokres1 + eallokres2 + c43dmd_met_by_lokres
     $    + c44dmd_met_by_lokres - amin1(DMDS236,0.0)
     $    + chgrnfistapoga(jday) + chgrnfnls(jday) + uisrnff(jday)
         vol_flow_res =  -amin1(eqs308,0.0)-amax1(KFLO(IFLS3)
```

```
     $,0.0)*1.9835-amax1(KFLO(IFLS2),0.0)*1.9835-DMDS4-AMAX1(DMDS236
     $,0.0)-dmdsemind(jday)- ws_to_roten_frm_lok*1.9835 - fplresws(jday)
     $    -KFLO(IFLHG5)*1.9835-amax1(KFLO(IFLAGQ),0.0)*1.9835
     $    -KFLO(iflc10a)*1.9835 - eallokres1 - eallokres2
     $    -c43dmd_met_by_lokres - c44dmd_met_by_lokres
     $    - chgdmdistapoga(jday) - chgdmdnls(jday) - chgdmdnels(jday) +
     $    chgrnfnels(jday)
c
         s354_res = amax1(KFLO(IFLS3),0.0)
         s351_res = amax1(KFLO(IFLS2),0.0)
         s352_res = KFLO(IFLHG5)
         dmnds4_res = amax1(dmds4,0.0)/1.9835
         dmnds236_res = amax1(dmds236,0.0)/1.9835
         dmndsemind_res = dmdsemind(jday)/1.9835
         rlokreswsroten = ws_to_roten_frm_lok
         flpresws_res = fplresws(jday)/1.9835
         r298dws_res = amax1(KFLO(IFLAGQ),0.0)
         l8ws_res = amax1(KFLO(iflc10a),0.0)
       endif
       S1 = SPREV + vol_flow_litzone
      endif
C
      return
      end
```

lake_reg_wca.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lake_reg_wca.F,v 1.23 2005/06/10 13:29:10 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lake_reg_wca.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE CALCULATES EXCESS WATER FROM LAKE OKEECHOBEE
C   TO APPROPRIATE DESTINATION(S). DESTINATIONS CAN INCLUDE
C   RESERVOIRS,ASRs,AND WATER CONSERVATION AREAS(WCAs).
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lake_reg_wca (volwtr, finstor, finstor_res, numd,
     & dmds4, dmds236, s65ernff, dmdsemind, jday,divers,
     & lo_reg_zone, lok_supp_day_c43est, bflo_cap_c43est, runoff_south,
     & lok_supp_day_c44est, bflo_cap_c44est, ifacteaa, stgref_asr,
     & stgref, stgref_asr_rec, hres01, hres02, qin_max01, qin_max02,
     & ares01, ares02, supp_day_c43est_tmp, supp_day_c44est_tmp,
     & rlokreg2res1, rlokreg2res2, reg2est1, reg2est2, flow_asr_inj,
     & prev_avg_lokstg, avg_lokstage, max_days_mean, rf_factor,
     & rmetvol_res, no_of_hurr_pred, climate_index, norstorrf,
     & norstorpet, tcrnbsres_to_lok, ws_to_roten_frm_lok, fplresws,
     & res_seep_factor, tot_cnl_dmd_asr, ro_factor, tcnsrnff, sprev_res,
     & s1, dmnd_lec_s5a, dmnd_lec_s6, dmnd_lec_sa2, stg_backflw_stl,
     & stgref_pulse_bot, stgref_nstor_to_lok, stgref_lok_to_nstor,
     & stlro, calro, vol_flow_res, runoff_south_bef_divers, rstgpls,
     & stgref_lok_to_eaasurgt, stgref_regtoest, kfloref, dellec,
     & ibeg_mth_lokdev, iend_mth_lokdev, ibeg_day_lokdev,
     & iend_day_lokdev, iregcals79opt, rmthly_rf_lok, cfsoth1,
     & cfsoth_litz1, cfsoth_res1, res_struc_inflw, day_rain_nsm,
     & isum_nsm,minx_nsm, maxx_nsm, rf_factor_lok, ro_factor_lok,
     & iopt_drawdown, ws_to_sta5e, s4bsn_to_lok, uisrnff,
     & stgref_state_eaares,rlokreg_to_l8,capacc10a_bak)
```

```
 INCLUDE 'wmm_par.inc'
 INCLUDE 'abc.inc'
 INCLUDE 'agdata.inc'
 INCLUDE 'et.inc'
 INCLUDE 'c1.inc'
 INCLUDE 'c2.inc'
 INCLUDE 'routc.inc'
 INCLUDE 'wrout.inc'
 INCLUDE 'stat.inc'
 INCLUDE 'stas.inc'
 INCLUDE 'simflows.inc'
 INCLUDE 'lakwca.inc'
 INCLUDE 'cstlucie.inc'
 INCLUDE 'ccaloos.inc'
 INCLUDE 'asr_param.inc'
 INCLUDE 'asr1.inc'
 INCLUDE 'asr2.inc'
 INCLUDE 'resadj.inc'
 INCLUDE 'eaa_caps.inc'
 INCLUDE 'resws.inc'
 INCLUDE 'opersched.inc'
 INCLUDE 'loktarg.inc'
 INCLUDE 'wcasim.inc'
 INCLUDE 'lokres.inc'
 INCLUDE 'climvar.inc'

 DIMENSION capac_bot_zone(2), capac_zone(2), climate_index(12),
& dmdsemind(366),
& flow_asr_inj(max_n_eaa_condt,max_n_reg_rel_basin), fplresws(366),
& inflow_indx(max_n_reg_rel_basin), isum_nsm(max_n_rows_incl_lok),
& maxx_nsm(max_n_rows_incl_lok), maxxlok(max_n_rows_lok),
& minx_nsm(max_n_rows_incl_lok), minxlok(max_n_rows_lok),
& minxlokres(max_n_rows_lok), numd(12),
& rem_weir_capac(max_n_eaa_condt), res_struc_inflw(max_n_res),
& rf_factor(12), rf_factor_lok(12), ro_factor(12),
& ro_factor_lok(12), rstgpls(max_n_puls_rel),
& runoff_south(max_n_eaa_condt),
& runoff_south_bef_divers(max_n_eaa_condt), s65ernff(366),
& stg_limit_reg(max_n_eaa_condt), tcnsrnff(366), volwtr(max_n_res),
& weircff_reg(max_n_eaa_condt,max_n_reg_rel_basin),
& zoneid_p(max_n_lok_zones), uisrnff(366)

 CHARACTER ceverg_needs(max_n_eaa_condt)*3, clokzone*7,
& clokzone_wca*7, cmulti_seas_forec*7, cmulti_seas_forec_wca*7,
& cseas_forec*7, cseas_forec_wca*7, cstg_lok_pres*20,
& ctrib_hydro_cond*7, ctrib_hydro_cond_wca*7,
& reg_lim_indic(max_n_eaa_condt)*6, reg_lim_operhyd*6, zoneid_p*7,
& zoneid_tmp*7

 LOGICAL lregtres

 REAL bflo_cap_c43est, bflo_cap_c44est, darefet,
& day_rain_nsm(max_n_cells_nsm,31), kfloref(max_n_eaa_condt),
& lketvol, lketvol_res, lok_supp_day_c43est, lok_supp_day_c44est,
& lok2res, loketlitz, loketres, loketvol, norstorpet(366),
& norstorrf(366)
```

```
      DATA minxlok /16,14,13,11,9,8,7,7,7,8,10,11,12,13,14,15,16/
      DATA minxlokres /16,14,13,11,17,16,15,14,14,15,16,17,18,18,18,18,
     & 18/
      DATA maxxlok /19,20,20,20,20,21,22,22,22,22,22,21,21,21,20,19,18/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   REGULATORY RELEASES

      minylok = 55
      maxylok = 71
      reg2est1 = 0.0
      reg2est2 = 0.0
      total_divers_to_eaares = 0.0
      cseas_forec = 'NOT_APP'
      cmulti_seas_forec = 'NOT_APP'
      ctrib_hydro_cond = 'NOT_APP'
      cstg_lok_pres = 'NOT_APPLICABLE      '
      cseas_forec_wca = 'NOT_APP'
      cmulti_seas_forec_wca = 'NOT_APP'
      ctrib_hydro_cond_wca = 'NOT_APP'
      do i = 1,max_n_eaa_condt
       ceverg_needs(i) = 'NA '
      enddo
      do i = 1,max_n_eaa_condt
       reg_lim_indic(i) = 'NA '
      enddo
      do i = 1,max_n_lok_zones
       zoneid_tmp = zoneid(i)
       do k = 1,LEN(zoneid_tmp)
        if (zoneid_tmp(k:k).ne.' ') then
         j = k
        endif
       enddo
       do k = 1,j
        if (zoneid_tmp(k:k).eq.' ') then
         zoneid_tmp(k:k) = '_'
        endif
       enddo
       zoneid_p(i) = zoneid_tmp
      enddo
c
c  compute volume of rainfall over LOK (acre-ft)
c
      rainlok = 0.0
      area_lok_rain = 0.0
      do iylok = minylok,maxylok
         do ixlok = minxlok(iylok-minylok+1),maxxlok(iylok-minylok+1)
            nodelok = ixlok - minx_nsm(iylok) + 1 + isum_nsm(iylok)
            rainlok = rainlok + day_rain_nsm(nodelok,iday)*gdar
     $      /(12.*43560.)
            area_lok_rain = area_lok_rain + gdar/43560.
         enddo
      enddo
c
c   compute rainfall over reservoir portion of LOK
```

```
c
      if (lok_split_option .eq. 1) then
        rainlok_res = 0.0
        do iylok = minylok,maxylok
          do ixlok =minxlokres(iylok-minylok+1),maxxlok(iylok-minylok+1)
             nodelokres = ixlok - minx_nsm(iylok) + 1 + isum_nsm(iylok)
             rainlok_res = rainlok_res + day_rain_nsm(nodelokres,iday)
     $       * gdar/(12.0*43560.)
          enddo
        enddo
        rainlok_litz = rainlok - rainlok_res
      endif
c
      jday_dev_beg = julian(ibeg_mth_lokdev,ibeg_day_lokdev,numd)
      jday_dev_end = julian(iend_mth_lokdev,iend_day_lokdev,numd)
      if (jday .ge. jday_dev_beg .and. jday .lt. jday_dev_end)
     $   then
         ioper_opt = 2
      else
         ioper_opt = 1
      endif
c
      if (iregcals79opt .eq. 1) then
         cal_rnff_frm_reg = calro
      else
         cal_rnff_frm_reg = 0.0
      endif
c
      call match ('EAARSN',1,6,resname,ntotres+1,6,ieaarsn_no)
      call match ('EAARSS',1,6,resname,ntotres+1,6,ieaarss_no)
      if (ieaarsn_no .eq. 0) ieaarsn_no = ntotres+1
      if (ieaarss_no .eq. 0) ieaarss_no = ntotres+1
      do kk = 1,nmtr
        if (ieaarsn_no .le. ntotres) then
          IF (IBSN(MTRP(kk,1)) .EQ. ibsn_no_res(ieaarsn_no)) then
            stg_eaa_res_2a = acstg(kk,1)
          ENDIF
        endif
        if (ieaarss_no .le. ntotres) then
          if (IBSN(MTRP(kk,1)) .EQ. ibsn_no_res(ieaarss_no)) then
            stg_eaa_res_2b = acstg(kk,1)
          endif
        endif
      enddo
c
      if (lok_split_option .eq. 0) then
       stagelo_after_ws = STG2STO(S1,1)
       AVLKSTORG = S1 - STG2STO(stgref,0)
       AVLKSTOR_ASR = S1 - STG2STO(stgref_asr,0)
       AVLKSTOR_EAASRGT = S1 - STG2STO(stgref_lok_to_eaasurgt,0)
       AVLKSTOR_ABOVE_LOK_REG_TO_EST = S1 - STG2STO(stgref_regtoest,0)
       avlkstor_state_eaares = S1 - STG2STO(stgref_state_eaares,0)
C
       IF (AVLKSTOR_ASR .LE. 0.0) THEN
         RLKSTOR = S1
       ELSE
         RLKSTOR = STG2STO(stgref_asr,0)
```

```
        ENDIF
       else
        stagelo_after_ws = STG2STOLITZ(S1,1)
        AVLKSTORG = S1 - STG2STOLITZ(stgref,0)
        AVLKSTOR_ASR = S1 - STG2STOLITZ(stgref_asr,0)
        avlkstor_state_eaares = S1 - STG2STOLITZ(stgref_state_eaares,0)
        AVLKSTOR_EAASRGT = S1 - STG2STOLITZ(stgref_lok_to_eaasurgt,0)
        AVLKSTOR_ABOVE_LOK_REG_TO_EST = S1-STG2STOLITZ(stgref_regtoest,0)
C
        IF (AVLKSTOR_ASR .LE. 0.0) THEN
          RLKSTOR = S1
        ELSE
          RLKSTOR = STG2STOLITZ(stgref_asr,0)
        ENDIF
       endif
       rlkstg = stagelo_after_ws
C
       S2 =  AMAX1(AVLKSTORG, 0.0)
       S2_ASR = AMAX1(AVLKSTOR_ASR,0.0)
       S2_REGTOEST = AMAX1(AVLKSTOR_ABOVE_LOK_REG_TO_EST,0.0)
       S2_EAARRSURGT = AMAX1(AVLKSTOR_EAASRGT,0.0)
       S2_state_eaares = AMAX1(avlkstor_state_eaares,0.0)
C
C   INJECT WATER FROM LAKE OKEECHOBEE INTO ASR RESERVOIRS
C
       do i = 1,neaabsn
        do k = 1,no_of_res_reg_frm_lok(i)
          flow_asr_inj(i,k) = 0.0
        enddo
       enddo
c
c
       call bpts(IDAY,MONTH,stagelo_after_ws,1,mzone,0,stgrefa)
       call bpts(iday,month,10.0,1,n_fc_zones+12,1,zonec_norm)
C
C  ROUTING EXCESS WATER FROM LOK INTO PROP RESERVOIR(S)
C  OPTION TO PRIORTIZE : HAVE RESERVOIR AS MAIN
C  RECIPIENT OF LOK EXCESS WATER IF RESERVOIR ID DESIGNED
C  FOR SUCH PURPOSE(iprop_res_prior_opt_lokreg = 1). THUS IF
C  RESERVOIR IS PROIRITY,THEN VOLUME OF EXCESS WATER
C  FROM LOK IS SUBJECT TO CANAL CONVEYANCE WITH PUMPED
C  FLOW INTO RESERVOIR WITH INTAKE LEVELS AT PUMP THE SAME
C  AS INTAKE LEVEL AT INFLOW PUMP INTO APPROPRIATE STA.
C
c  compute EAA conveyance capacities for neutral case for
c  Miami and NNR canals with pumped flow
c
       if (iprop_res_prior_opt_lokreg .eq. 1) then
c
         if (istaopt (is8bsn-6) .eq. 0) then
c  Miami Canal
           ireach = 1
           istype = 1
           istr_eaa = 5
           chw = stagelo
           ctw = pmp_int(is8bsn-6)
           delta0 = (chw - ctw) / 25.0
```

```
          call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo
     $         ,delta0,1.0,100,miamic_cap_upper_pump,s354cap,s354_tw
     $         ,ifacteaa)
          if (s354_tw .gt. rmaxstg(is8bsn-6)) then
            s354_tw = rmaxstg(is8bsn-6)
            s354cap = -1.0
            call eaaspill(s354cap,stagelo,rmaxstg(is8bsn-6),5
     $              ,ifacteaa)
            qcanal = -1.0
            if (s354_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,s354_tw,ctw,
     &         ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            miamic_cap_upper_pump = qcanal
          endif
          grf3amx = miamic_cap_upper_pump
        else
          istype = 1
          istr_eaa = 5
          chw = stagelo
          ctw = pmp_int_new(is8bsn-6)
          delta0 = (chw - ctw) / 25.0
          ireach = 5
          call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo
     $           ,delta0,1.0,100,miamic_cap_upper_pump,s354cap,s354_tw
     $           ,ifacteaa)
          if (s354_tw .gt. rmaxstg(is8bsn-6)) then
            s354_tw = rmaxstg(is8bsn-6)
            s354cap = -1.0
            call eaaspill(s354cap,stagelo,rmaxstg(is8bsn-6),5,ifacteaa)
            qcanal = -1.0
            if (s354_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,s354_tw,ctw,
     $         ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            miamic_cap_upper_pump = qcanal
          endif
          grf3amx = miamic_cap_upper_pump
        endif
        s354mx = s354cap
c
c  NNR Canal
c
         if (istaopt(is7bsn-6) .eq. 0) then
          ireach  = 2
          istype = 1
          istr_eaa = 6
          chw = stagelo
          ctw = pmp_int(is7bsn-6)
          delta0 = (chw - ctw) / 25.0
          call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo
     $         ,delta0,1.0,100,nnrc_cap_upper_pump,s351cap
     $         ,s351_tw,ifacteaa)
```

```
          if (s351_tw .gt. rmaxstg(is7bsn-6)) then
            s351_tw = rmaxstg(is7bsn-6)
            s351cap = -1.0
            call eaaspill(s351cap,stagelo,rmaxstg(is7bsn-6),6
     $           ,ifacteaa)
            qcanal = -1.0
            if (s351_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,s351_tw,ctw,
     &         ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            nnrc_cap_upper_pump = qcanal
          endif
          fllks6 = -1.0
          call eaa_prof_lu_table(fllks6,s351_tw,pmp_int(is6bsn-6)
     $         ,3,ifacteaa)
          grf2amx = nnrc_cap_upper_pump
        else
          ireach = 6
          istype = 1
          istr_eaa = 6
          chw = stagelo
          ctw = pmp_int_new(is7bsn-6)
          delta0 = (chw - ctw) / 25.0
          call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo
     $           ,delta0,1.0,100,nnrc_cap_upper_pump,s351cap,s351_tw
     $           ,ifacteaa)
          if (s351_tw .gt. rmaxstg(is7bsn-6)) then
            s351_tw = rmaxstg(is7bsn-6)
            s351cap = -1.0
            call eaaspill(s351cap,stagelo,rmaxstg(is7bsn-6),6,ifacteaa)
            qcanal = -1.0
            if (s351_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,s351_tw,ctw,
     &           ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            nnrc_cap_upper_pump = qcanal
          endif
          fllks6 = -1.0
          call eaa_prof_lu_table(fllks6,s351_tw,pmp_int(is6bsn-6)
     $          ,3,ifacteaa)
          grf2amx = nnrc_cap_upper_pump
        endif
        s351mx = s351cap
      endif
c
      do k = 1,max_n_reg_rel_basin
        inflow_indx(k) = 1
      enddo
      DO 7000 IS = 1,NSTRCA
         GOTO (2000,3000,4000,5000,6000) IRG_TO_RES_PRTY(IS)
 2000    if (is5absn-6 .ge. 2) then
           do k = 1,no_of_res_reg_frm_lok(is5absn-6)
             do isindx = is5absn-6,2,-1
```

```
         if (ieaa_res_asr_reg(is5absn-6,k) .eq. ieaa_res_asr_reg
$        (isindx-1,k)) then
           inflow_indx(k) = inflow_indx(k) + 1
         endif
        enddo
      enddo
    endif
     ratio_to_res= rcpfacteaa_res(is5absn-6)
$                     / rcpfacteaa_canal(is5absn-6)
     do k = 1,no_of_res_reg_frm_lok(is5absn-6)
      if (ieaa_res_asr_reg(is5absn-6,k).eq.ieaa_res_asr(is5absn-6))
$           then
         weircff_reg(is5absn-6,k) = rem_weir_capac(is5absn-6)
      else
         weircff_reg(is5absn-6,k) = weircff(ieaa_res_asr_reg
$                (is5absn-6,k),inflow_indx(k),1)
      endif
      if (opt_sim_lok_eaares .ne. 'EXCESS') then
         if (iclimate_opt .eq. 1) then
            pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
$                              - totloketvol3(month)
$                              - avg_demand_lok3(month)
            month_pred = month + 3
            if (month_pred .gt. 12) month_pred = month_pred - 12
            call bpts(iday,month_pred,10.0,1,n_fc_zones+3,1
$                     ,eaares_surge_pred)
            if (lok_split_option .eq. 0) then
              SPRES = STG2STO(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STO(STO_PROJ,1)
            else
              SPRES = STG2STOLITZ(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
            endif
            if (opt_for_reg_rel_to_res(is5absn-6,k) .eq. 'FOREC')
$               then
              lregtres = (pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 1
$               .and. (trib_rf_et .gt.
$               trib_rfet_thres .or. s65e_runff_wkly
$               .ge. s65e_runff_wkly_thres))
$               .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
$               mzone_puls_top .and. mzone .le. mzone_puls_bot
$               .and. (stagelo .lt. stgref_lok_to_eaasurgt
$               .and.climate_index(month) .gt. s_forec_eaar_thres2
$               .or. (stagelo.ge.stgref_lok_to_eaasurgt
$               .and. climate_index(month) .gt. s_forec_eaar_thres1
$               )).and. izone_flex_eaares_opt .eq. 1))
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or. (nint(S2_state_eaares) .gt. 0 .and.
$               izone_flex_eaares_opt .eq. 0
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or.(pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 0
$               .and.izone_flex_eaares_opt.eq.0)
            else
```

```
              lregtres = (nint(S2_state_eaares).gt. 0.and.
$                 izone_flex_eaares_opt.eq.0) .or.(stagelo .gt.
$                 stgref_lok_to_eaasurgt .and.izone_flex_eaares_opt
$                 .eq.1)
            endif
            if (lregtres) then
              AVCAP_S352  = S352MX*ratio_to_res - KFLO(IFLHG5)
              if (iconv_use_inj(is5absn-6) .eq. 0) then
                vol_to_inject =amin1(AVCAP_S352
$               ,volwtr(ieaa_res_asr_reg(is5absn-6,k))
$               ,weircff_reg(is5absn-6,k))
              else
                avail_inj = amax1(fllkwpb*ratio_to_res
&                           -kfloref(is5absn-6),0.0)
                vol_to_inject =amin1(AVCAP_S352
$                ,volwtr(ieaa_res_asr_reg(is5absn-6,k)),avail_inj)
                kfloref(is5absn-6) = kfloref(is5absn-6)
$                      +  vol_to_inject
              endif
              flow_to_eaa_res_reg(is5absn-6,k) = vol_to_inject
$              + flow_to_eaa_res_reg(is5absn-6,k)
              volwtr(ieaa_res_asr_reg(is5absn-6,k))= volwtr
$              (ieaa_res_asr_reg(is5absn-6,k))-vol_to_inject
              S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
              S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
              S2_ASR = S2_ASR - vol_to_inject*1.9835
              S2     = S2     - vol_to_inject*1.9835
              S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
              S2     = AMAX1(S2,0.0)
              S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
              S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
              S2_state_eaares = AMAX1(S2_state_eaares,0.0)
              KFLO(IFLHG5) = KFLO(IFLHG5) + vol_to_inject
              flow_asr_inj(is5absn-6,k) = vol_to_inject
              total_divers_to_eaares = vol_to_inject
$                      + total_divers_to_eaares
            else
              flow_asr_inj(is5absn-6,k) = 0.0
            endif
          else
             avail_vol_to_res = S2_EAARRSURGT
          endif
        else
          if (iasr_opt_eaa(is5absn-6) .eq. 1) then
            avail_vol_to_res = S2_EAARRSURGT
          else
            avail_vol_to_res = S2_state_eaares
          endif
        endif
      ntarg = 0
      do itarg = 1,no_targ_loc(is5absn-6)
        if (LWSLO(itarg_area(itarg,is5absn-6))) then
          ntarg = ntarg + 1
        endif
      enddo
      IF (avail_vol_to_res .GT. 0.0 .and. (iclimate_opt .eq. 0
$          .or. opt_sim_lok_eaares .eq. 'EXCESS')) then
```

```
            IF (nint(S2) .GT. 0 .OR. (avg_lokstage .ge. prev_avg_lokstg
     $       .AND.max_days_mean.gt.0) .OR. (nint(S2) .EQ. 0 .AND.
     $       nint(dmnd_lec_s5a) .eq. 0 .and. ntarg .eq. 0)) then
             AVCAP_S352  = S352MX*ratio_to_res - KFLO(IFLHG5)
             if (iconv_use_inj(is5absn-6) .eq. 0) then
               vol_to_inject =amin1(avail_vol_to_res/1.9835,AVCAP_S352
     $          ,volwtr(ieaa_res_asr_reg(is5absn-6,k))
     $          ,weircff_reg(is5absn-6,k))
             else
               avail_inj = amax1(fllkwpb*ratio_to_res
     &                   - kfloref(is5absn-6),0.0)
               vol_to_inject =amin1(avail_vol_to_res/1.9835,AVCAP_S352
     $               ,volwtr(ieaa_res_asr_reg(is5absn-6,k)),avail_inj)
               kfloref(is5absn-6) = kfloref(is5absn-6) + vol_to_inject
             endif
             flow_to_eaa_res_reg(is5absn-6,k) = vol_to_inject
     $       + flow_to_eaa_res_reg(is5absn-6,k)
             volwtr(ieaa_res_asr_reg(is5absn-6,k))= volwtr
     $        (ieaa_res_asr_reg(is5absn-6,k))-vol_to_inject
             S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
             S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
             S2_ASR = S2_ASR - vol_to_inject*1.9835
             S2     = S2     - vol_to_inject*1.9835
             S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
             S2     = AMAX1(S2,0.0)
             S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
             S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
             S2_state_eaares = AMAX1(S2_state_eaares,0.0)
             KFLO(IFLHG5) = KFLO(IFLHG5) + vol_to_inject
             flow_asr_inj(is5absn-6,k) = vol_to_inject
             total_divers_to_eaares = vol_to_inject
     $                 + total_divers_to_eaares
            ENDIF
           ENDIF
           ENDDO
          GO TO 7000
3000      ratio_to_res= rcpfacteaa_res(is6bsn-6)
     $                  / rcpfacteaa_canal(is6bsn-6)
          do k = 1,no_of_res_reg_frm_lok(is6bsn-6)
            do isindx = is6bsn-6,2,-1
             if (ieaa_res_asr_reg(is6bsn-6,k) .eq. ieaa_res_asr_reg
     $       (isindx-1,k)) then
               inflow_indx(k) = inflow_indx(k) + 1
             endif
            enddo
          enddo
          do k = 1,no_of_res_reg_frm_lok(is6bsn-6)
           if (ieaa_res_asr_reg(is6bsn-6,k) .eq. ieaa_res_asr(is7bsn-6))
     $           then
              weircff_reg(is6bsn-6,k) = rem_weir_capac(is7bsn-6)
           else
              weircff_reg(is6bsn-6,k) = weircff(ieaa_res_asr_reg
     $           (is6bsn-6,k),inflow_indx(k),1)
           endif
           if (opt_sim_lok_eaares .ne. 'EXCESS') then
              if (iclimate_opt .eq. 1) then
                 pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
```

```
$                              - totloketvol3(month)
$                              - avg_demand_lok3(month)
            month_pred = month + 3
            if (month_pred .gt. 12) month_pred = month_pred - 12
            call bpts(iday,month_pred,10.0,1,n_fc_zones+3,1
$                     ,eaares_surge_pred)
            if (lok_split_option .eq. 0) then
              SPRES = STG2STO(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STO(STO_PROJ,1)
            else
              SPRES = STG2STOLITZ(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
            endif
            if (opt_for_reg_rel_to_res(is6bsn-6,k) .eq. 'FOREC')
$               then
              lregtres = (pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 1
$               .and. (trib_rf_et .gt.
$               trib_rfet_thres .or. s65e_runff_wkly
$               .ge. s65e_runff_wkly_thres))
$               .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
$               mzone_puls_top .and. mzone .le. mzone_puls_bot
$               .and. (stagelo .lt. stgref_lok_to_eaasurgt
$               .and.climate_index(month) .gt. s_forec_eaar_thres2
$               .or. (stagelo.ge.stgref_lok_to_eaasurgt
$               .and. climate_index(month) .gt. s_forec_eaar_thres1
$               )).and. izone_flex_eaares_opt .eq. 1))
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or. (nint(S2_state_eaares) .gt. 0 .and.
$               izone_flex_eaares_opt .eq. 0
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or.(pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 0
$               .and.izone_flex_eaares_opt.eq.0)
            else
              lregtres = (nint(S2_state_eaares).gt. 0.and.
$                 izone_flex_eaares_opt.eq.0) .or.(stagelo .gt.
$                 stgref_lok_to_eaasurgt .and.izone_flex_eaares_opt
$                 .eq.1)
            endif
            if (lregtres) then
              if (iconv_use_inj(is6bsn-6) .eq. 0) then
                vol_to_inject =amin1(s351mx*ratio_to_res
$                        - kflo(ifls2)
$                 ,volwtr(ieaa_res_asr_reg(is6bsn-6,k))
$                 ,weircff_reg(is6bsn-6,k))
              else
                avail_inj =fllks6*ratio_to_res - kfloref(is6bsn-6)
                avail_inj = amax1(avail_inj,0.0)
                vol_to_inject=amin1(s351mx*ratio_to_res-kflo(ifls2)
$                 ,volwtr(ieaa_res_asr_reg(is6bsn-6,k))
$                 ,weircff_reg(is6bsn-6,k),avail_inj)
                kfloref(is6bsn-6) = kfloref(is6bsn-6)
$                         + vol_to_inject
              endif
```

```
                flow_to_eaa_res_reg(is6bsn-6,k) = vol_to_inject
     $            + flow_to_eaa_res_reg(is6bsn-6,k)
                weircff_reg(is6bsn-6,k)= weircff_reg(is6bsn-6,k)
     $            - vol_to_inject
                if (ieaa_res_asr_reg(is6bsn-6,k).eq.
     $            ieaa_res_asr(is7bsn-6)) then
                  rem_weir_capac(is7bsn-6) = rem_weir_capac(is7bsn-6)
     $                - vol_to_inject
                endif
                volwtr(ieaa_res_asr_reg(is6bsn-6,k))=volwtr
     $             (ieaa_res_asr_reg(is6bsn-6,k))-vol_to_inject
                S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
                S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
                S2_ASR = S2_ASR - vol_to_inject*1.9835
                S2     = S2     - vol_to_inject*1.9835
                S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
                S2     = AMAX1(S2,0.0)
                S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
                S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
                S2_state_eaares = AMAX1(S2_state_eaares,0.0)
                kflo(ifls2) = kflo(ifls2) + vol_to_inject
                flow_asr_inj(is6bsn-6,k) = vol_to_inject
                total_divers_to_eaares = vol_to_inject
     $                + total_divers_to_eaares
              else
                flow_asr_inj(is6bsn-6,k) = 0.0
              endif
            else
              avail_vol_to_res = S2_EAARRSURGT
            endif
          else
            IF (iasr_opt_eaa(is7bsn-6) .eq. 1) then
              avail_vol_to_res = S2_EAARRSURGT
            else
              avail_vol_to_res = S2_state_eaares
            endif
          endif
c
          ntarg = 0
          do itarg = 1,no_targ_loc(is6bsn-6)
            if (LWSLO(itarg_area(itarg,is6bsn-6))) then
              ntarg = ntarg + 1
            endif
          enddo
c
          IF (avail_vol_to_res .GT. 0.0 .and. (iclimate_opt .eq. 0
     $          .or. opt_sim_lok_eaares .eq. 'EXCESS')) then
           IF (nint(S2) .GT. 0 .OR. (avg_lokstage .ge. prev_avg_lokstg
     $          .AND.max_days_mean.gt.0) .OR. (S2 .EQ. 0.0 .AND.
     $          nint(dmnd_lec_s6) .eq. 0 .and. ntarg .eq. 0) .and.
     $          kflo(ifls2) .ge. 0.0) then
            if (iconv_use_inj(is6bsn-6) .eq. 0) then
             vol_to_inject =amin1(s351mx*ratio_to_res - kflo(ifls2)
     $          ,avail_vol_to_res/1.9835,volwtr(ieaa_res_asr_reg
     $          (is7bsn-6,k)),weircff_reg(is7bsn-6,k))
            else
             avail_inj = fllks6*ratio_to_res - kfloref(is6bsn-6)
```

```
            avail_inj = amax1(avail_inj,0.0)
            vol_to_inject = amin1(s351mx*ratio_to_res - kflo(ifls2)
     $        ,avail_vol_to_res/1.9835,volwtr(ieaa_res_asr_reg
     $        (is7bsn-6,k)),weircff_reg(is7bsn-6,k),avail_inj)
            kfloref(is6bsn-6) = kfloref(is6bsn-6) + vol_to_inject
           endif
           flow_to_eaa_res_reg(is6bsn-6,k) = vol_to_inject
     $       + flow_to_eaa_res_reg(is6bsn-6,k)
           weircff_reg(is6bsn-6,k)= weircff_reg(is6bsn-6,k)
     $       - vol_to_inject
           if (ieaa_res_asr_reg(is6bsn-6,k).eq.ieaa_res_asr(is7bsn-6))
     $         then
            rem_weir_capac(is7bsn-6) = rem_weir_capac(is7bsn-6)
     $                  - vol_to_inject
           endif
          volwtr(ieaa_res_asr_reg(is6bsn-6,k))=volwtr(ieaa_res_asr_reg
     $        (is6bsn-6,k))-vol_to_inject
           S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
           S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
           S2_ASR = S2_ASR - vol_to_inject*1.9835
           S2     = S2     - vol_to_inject*1.9835
           S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
           S2     = AMAX1(S2,0.0)
           S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
           S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
           S2_state_eaares = AMAX1(S2_state_eaares,0.0)
           kflo(ifls2) = kflo(ifls2) + vol_to_inject
           flow_asr_inj(is6bsn-6,k) = vol_to_inject
           total_divers_to_eaares = vol_to_inject
     $                  + total_divers_to_eaares
          ENDIF
         ENDIF
        ENDDO
        GO TO 7000
4000    if (is7bsn-6 .ge. 2) then
          do k = 1,no_of_res_reg_frm_lok(is7bsn-6)
            do isindx = is7bsn-6,2,-1
             if (ieaa_res_asr_reg(is7bsn-6,k) .eq.
     $        ieaa_res_asr_reg(isindx-1,k)) then
              inflow_indx(k) = inflow_indx(k) + 1
             endif
            enddo
          enddo
        endif
         ratio_to_res= rcpfacteaa_res(is7bsn-6)
     $                / rcpfacteaa_canal(is7bsn-6)
         do k = 1,no_of_res_reg_frm_lok(is7bsn-6)
          if (ieaa_res_asr_reg(is7bsn-6,k) .eq. ieaa_res_asr(is7bsn-6))
     $          then
             weircff_reg(is7bsn-6,k) = rem_weir_capac(is7bsn-6)
          else
             weircff_reg(is7bsn-6,k) = weircff(ieaa_res_asr_reg
     $           (is7bsn-6,k),inflow_indx(k),1)
          endif
          if (opt_sim_lok_eaares .ne. 'EXCESS') then
             if (iclimate_opt .eq. 1) then
                pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
```

```
$                           - totloketvol3(month)
$                           - avg_demand_lok3(month)
            month_pred = month + 3
            if (month_pred .gt. 12) month_pred = month_pred - 12
            call bpts(iday,month_pred,10.0,1,n_fc_zones+3,1
$                     ,eaares_surge_pred)
            if (lok_split_option .eq. 0) then
              SPRES = STG2STO(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STO(STO_PROJ,1)
            else
              SPRES = STG2STOLITZ(rlkstg,0)
              STO_PROJ = SPRES + pred_change_in_lok_sto
              pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
            endif
            if (opt_for_reg_rel_to_res(is7bsn-6,k) .eq. 'FOREC')
$               then
              lregtres = (pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 1
$               .and. (trib_rf_et .gt.
$               trib_rfet_thres .or. s65e_runff_wkly
$               .ge. s65e_runff_wkly_thres))
$               .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
$               mzone_puls_top .and. mzone .le. mzone_puls_bot
$               .and. (stagelo .lt. stgref_lok_to_eaasurgt
$               .and.climate_index(month) .gt. s_forec_eaar_thres2
$               .or. (stagelo.ge.stgref_lok_to_eaasurgt
$               .and. climate_index(month) .gt. s_forec_eaar_thres1
$               )).and. izone_flex_eaares_opt .eq. 1))
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or. (nint(S2_state_eaares) .gt. 0 .and.
$               izone_flex_eaares_opt .eq. 0
$               .and. iadd_ops_to_eaa_res_line .eq. 1)
$               .or.(pred_lok_stg .gt. eaares_surge_pred
$               .and. itrib_hydro_cond_opt .eq. 0
$               .and.izone_flex_eaares_opt.eq.0)
            else
              lregtres = (nint(S2_state_eaares).gt. 0.and.
$                 izone_flex_eaares_opt.eq.0) .or.(stagelo .gt.
$                 stgref_lok_to_eaasurgt .and.izone_flex_eaares_opt
$                 .eq.1)
            endif
            if (lregtres) then
              if (iconv_use_inj(is7bsn-6) .eq. 0) then
                vol_to_inject=amin1(s351mx*ratio_to_res-kflo(ifls2)
$                ,volwtr(ieaa_res_asr_reg(is7bsn-6,k))
$                ,weircff_reg(is7bsn-6,k))
              else
                avail_inj = grf2amx*ratio_to_res-kfloref(is7bsn-6)
                avail_inj = amax1(avail_inj,0.0)
                vol_to_inject=amin1(s351mx*ratio_to_res-kflo(ifls2)
$                 ,volwtr(ieaa_res_asr_reg(is7bsn-6,k))
$                 ,weircff_reg(is7bsn-6,k)
$                 ,avail_inj)
                kfloref(is7bsn-6) = kfloref(is7bsn-6)
$                 + vol_to_inject
              endif
```

```
c
                  flow_to_eaa_res_reg(is7bsn-6,k) = vol_to_inject
     $              + flow_to_eaa_res_reg(is7bsn-6,k)
                  total_divers_to_eaares = vol_to_inject
     $                     + total_divers_to_eaares
c             print *,lregtres,vol_to_inject,weircff_reg(is7bsn-6,k)
c    $     ,s351mx,ratio_to_res,kflo(ifls2),grf2amx,kfloref,'  EAARN '
c    $     ,pred_lok_stg,eaares_surge_pred,S2,k,avail_inj
c    $     ,volwtr(ieaa_res_asr_reg(is7bsn-6,k))
c    $     ,resname(ieaa_res_asr_reg(is7bsn-6,k))
                  volwtr(ieaa_res_asr_reg(is7bsn-6,k))= volwtr
     $              (ieaa_res_asr_reg(is7bsn-6,k))-vol_to_inject
                  S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
                  S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
                  S2_ASR = S2_ASR - vol_to_inject*1.9835
                  S2     = S2     - vol_to_inject*1.9835
                  S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
                  S2     = AMAX1(S2,0.0)
                  S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
                  S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
                  S2_state_eaares = AMAX1(S2_state_eaares,0.0)
                  kflo(ifls2) = kflo(ifls2) + vol_to_inject
                  flow_asr_inj(is7bsn-6,k) = vol_to_inject
     $             + flow_asr_inj(is7bsn-6,k)
                else
                  flow_asr_inj(is7bsn-6,k) = 0.0
     $                  + flow_asr_inj(is7bsn-6,k)
                endif
              else
                avail_vol_to_res = S2_EAARRSURGT
              endif
            else
              IF (iasr_opt_eaa(is7bsn-6) .eq. 1) then
                avail_vol_to_res = S2_EAARRSURGT
              else
                avail_vol_to_res = S2_state_eaares
              endif
            endif
c
            ntarg = 0
            do itarg = 1,no_targ_loc(is7bsn-6)
              if (LWSLO(itarg_area(itarg,is7bsn-6))) then
                ntarg = ntarg + 1
              endif
            enddo
c
            IF (avail_vol_to_res .GT. 0.0 .and. (iclimate_opt .eq. 0
     $           .or. opt_sim_lok_eaares .eq. 'EXCESS')) then
             IF (nint(S2).GT. 0 .OR. (avg_lokstage .ge. prev_avg_lokstg
     $    .AND.max_days_mean.gt.0).OR.(nint(S2).EQ.0.AND.
     $    nint(dmnd_lec_sa2).eq. 0 .and. .not.(ntarg .gt. 0.and.
     $    nnrctwca3a.eq.'TRUE ')).and. kflo(ifls2) .ge. 0.0) then
              if (iconv_use_inj(is7bsn-6) .eq. 0) then
               vol_to_inject = amin1(s351mx*ratio_to_res - kflo(ifls2)
     $         ,avail_vol_to_res/1.9835,volwtr
     $         (ieaa_res_asr_reg(is7bsn-6,k)),weircff_reg(is7bsn-6,k))
              else
```

```
              avail_inj = grf2amx*ratio_to_res - kfloref(is7bsn-6)
              avail_inj = amax1(avail_inj,0.0)
              vol_to_inject = amin1(s351mx*ratio_to_res - kflo(ifls2)
     $          ,avail_vol_to_res/1.9835,volwtr
     $          (ieaa_res_asr_reg(is7bsn-6,k)),weircff_reg(is7bsn-6,k)
     $          ,avail_inj)
c             print *,vol_to_inject,s351mx,ratio_to_res,
c    $        weircff_reg(is7bsn-6,k),avail_inj,avail_vol_to_res/1.9835,
c    $        'EAAREXCESS'
              kfloref(is7bsn-6) = kfloref(is7bsn-6) + vol_to_inject
             endif
c
              flow_to_eaa_res_reg(is7bsn-6,k) = vol_to_inject
     $          + flow_to_eaa_res_reg(is7bsn-6,k)
             volwtr(ieaa_res_asr_reg(is7bsn-6,k))=volwtr(ieaa_res_asr_reg
     $           (is7bsn-6,k))-vol_to_inject
              S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
              S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
              S2_ASR = S2_ASR - vol_to_inject*1.9835
              S2     = S2     - vol_to_inject*1.9835
              S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
              S2     = AMAX1(S2,0.0)
              S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
              S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
              S2_state_eaares = AMAX1(S2_state_eaares,0.0)
              kflo(ifls2) = kflo(ifls2) + vol_to_inject
              flow_asr_inj(is7bsn-6,k) = vol_to_inject
     $             + flow_asr_inj(is7bsn-6,k)
              total_divers_to_eaares = vol_to_inject
     $             + total_divers_to_eaares
            ENDIF
           ENDIF
          ENDDO
          GO TO 7000
 5000      ratio_to_res = rcpfacteaa_res(is8bsn-6)
     $                    /rcpfacteaa_canal(is8bsn-6)
           do k = 1,no_of_res_reg_frm_lok(is8bsn-6)
            if (ieaa_res_asr_reg(is8bsn-6,k) .eq. ieaa_res_asr(is8bsn-6))
     $            then
               weircff_reg(is8bsn-6,k) = rem_weir_capac(is8bsn-6)
            else
               weircff_reg(is8bsn-6,k) = weircff(ieaa_res_asr_reg
     $              (is8bsn-6,k),1,1)
            endif
            if (opt_sim_lok_eaares .ne. 'EXCESS') then
               if (iclimate_opt .eq. 1) then
                  pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
     $                              - totloketvol3(month)
     $                              - avg_demand_lok3(month)
                  month_pred = month + 3
                  if (month_pred .gt. 12) month_pred = month_pred - 12
                  call bpts(iday,month_pred,10.0,1,n_fc_zones+3,1
     $                      ,eaares_surge_pred)
                  if (lok_split_option .eq. 0) then
                    SPRES = STG2STO(rlkstg,0)
                    STO_PROJ = SPRES + pred_change_in_lok_sto
                    pred_lok_stg = STG2STO(STO_PROJ,1)
```

```
                  else
                    SPRES = STG2STOLITZ(rlkstg,0)
                    STO_PROJ = SPRES + pred_change_in_lok_sto
                    pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
                  endif
                  if (opt_for_reg_rel_to_res(is8bsn-6,k) .eq. 'FOREC')
     $                then
                    lregtres = (pred_lok_stg .gt. eaares_surge_pred
     $                .and. itrib_hydro_cond_opt .eq. 1
     $                .and. (trib_rf_et .gt.
     $                trib_rfet_thres .or. s65e_runff_wkly
     $                .ge. s65e_runff_wkly_thres))
     $                .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
     $                mzone_puls_top .and. mzone .le. mzone_puls_bot
     $                .and. (stagelo .lt. stgref_lok_to_eaasurgt
     $                .and.climate_index(month) .gt. s_forec_eaar_thres2
     $                .or. (stagelo.ge.stgref_lok_to_eaasurgt
     $                .and. climate_index(month) .gt. s_forec_eaar_thres1
     $                )).and. izone_flex_eaares_opt .eq. 1))
     $                .and. iadd_ops_to_eaa_res_line .eq. 1)
     $                .or. (nint(S2_state_eaares) .gt. 0 .and.
     $                izone_flex_eaares_opt .eq. 0
     $                .and. iadd_ops_to_eaa_res_line .eq. 1)
     $                .or.(pred_lok_stg .gt. eaares_surge_pred
     $                .and. itrib_hydro_cond_opt .eq. 0
     $                .and.izone_flex_eaares_opt.eq.0)
c                     print *,lregtres,pred_lok_stg,eaares_surge_pred
c    $                ,itrib_hydro_cond_opt,iadd_ops_to_eaa_res_line
c    $                ,izone_flex_eaares_opt,'   FOREC'
                  else
                    lregtres = (nint(S2_state_eaares).gt. 0.and.
     $                  izone_flex_eaares_opt.eq.0) .or.(stagelo .gt.
     $                  stgref_lok_to_eaasurgt .and.izone_flex_eaares_opt
     $                  .eq.1)
c                  print *,lregtres,S2_state_eaares,izone_flex_eaares_opt
c    $               ,stagelo,stgref_lok_to_eaasurgt,'   STATE  '
c    $               ,stgref_state_eaares
                  endif
                  if (lregtres) then
                    if (iconv_use_inj(is8bsn-6) .eq. 0) then
                      vol_to_inject=amin1(s354mx*ratio_to_res-kflo(ifls3)
     $                  ,volwtr(ieaa_res_asr_reg(is8bsn-6,k))
     $                  ,weircff_reg(is8bsn-6,k))
                    else
                      avail_inj = amax1(grf3amx*ratio_to_res
     $                     - kfloref(is8bsn-6),0.0)
                      vol_to_inject=amin1(s354mx*ratio_to_res-kflo(ifls3)
     $                 ,volwtr(ieaa_res_asr_reg(is8bsn-6,k))
     $                 ,weircff_reg(is8bsn-6,k),avail_inj)
                      kfloref(is8bsn-6) = kfloref(is8bsn-6)
     $                 + vol_to_inject
                    endif
c                 print *,lregtres,vol_to_inject,weircff_reg(is8bsn-6,k)
c    $     ,s354mx,ratio_to_res,kflo(ifls3),grf3amx,kfloref(is8bsn-6)
c    $     ,'  EAARM ',pred_lok_stg,eaares_surge_pred,S2,k,avail_inj
c    $     ,volwtr(ieaa_res_asr_reg(is8bsn-6,k)),iconv_use_inj(is8bsn-6)
c    $     ,resname(ieaa_res_asr_reg(is8bsn-6,k))
```

```
                flow_to_eaa_res_reg(is8bsn-6,k) = vol_to_inject
     $             + flow_to_eaa_res_reg(is8bsn-6,k)
                volwtr(ieaa_res_asr_reg(is8bsn-6,k))= volwtr
     $             (ieaa_res_asr_reg(is8bsn-6,k))-vol_to_inject
                S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
                S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
                S2_ASR = S2_ASR - vol_to_inject*1.9835
                S2     = S2     - vol_to_inject*1.9835
                S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
                S2     = AMAX1(S2,0.0)
                S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
                S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
                S2_state_eaares = AMAX1(S2_state_eaares,0.0)
                kflo(ifls3) = kflo(ifls3) + vol_to_inject
                flow_asr_inj(is8bsn-6,k) = vol_to_inject
                total_divers_to_eaares = vol_to_inject
     $                  + total_divers_to_eaares
              else
                flow_asr_inj(is8bsn-6,k) = 0.0
              endif
            else
              avail_vol_to_res = S2_EAARRSURGT
            endif
          else
            IF (iasr_opt_eaa(is8bsn-6) .eq. 1 ) then
              avail_vol_to_res = S2_EAARRSURGT
            else
              avail_vol_to_res = S2_state_eaares
            endif
          endif
c
          ntarg = 0
          do itarg = 1,no_targ_loc(is8bsn-6)
            if (LWSLO(itarg_area(itarg,is8bsn-6))) then
              ntarg = ntarg + 1
            endif
          enddo
c
        IF (avail_vol_to_res .GT. 0.0 .and. (iclimate_opt .eq. 0
     $          .or. opt_sim_lok_eaares .eq. 'EXCESS')) then
        IF (nint(S2) .GT. 0 .OR. (avg_lokstage .ge. prev_avg_lokstg
     $  .AND.max_days_mean.gt.0).OR.(nint(S2) .EQ. 0 .AND.
     $  nint(dmnd_lec_sa3).eq. 0 .and. ntarg .eq. 0)
     $  .and. kflo(ifls3) .ge. 0.0) then
           if (iconv_use_inj(is8bsn-6) .eq. 0) then
            vol_to_inject = amin1(s354mx*ratio_to_res - kflo(ifls3)
     $       ,avail_vol_to_res/1.9835,volwtr
     $       (ieaa_res_asr_reg(is8bsn-6,k)),weircff_reg(is8bsn-6,k))
           else
            avail_inj = amax1(grf3amx*ratio_to_res
     $                  - kfloref(is8bsn-6),0.0)
            vol_to_inject = amin1(s354mx*ratio_to_res - kflo(ifls3)
     $       ,avail_vol_to_res/1.9835,volwtr
     $       (ieaa_res_asr_reg(is8bsn-6,k)),weircff_reg(is8bsn-6,k)
     $       ,avail_inj)
            kfloref(is8bsn-6) = kfloref(is8bsn-6) + vol_to_inject
           endif
```

```
            flow_to_eaa_res_reg(is8bsn-6,k) = vol_to_inject
     $        + flow_to_eaa_res_reg(is8bsn-6,k)
           volwtr(ieaa_res_asr_reg(is8bsn-6,k))=volwtr(ieaa_res_asr_reg
     $          (is8bsn-6,k))-vol_to_inject
            S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
            S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
            S2_ASR = S2_ASR - vol_to_inject*1.9835
            S2     = S2     - vol_to_inject*1.9835
            S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
            S2     = AMAX1(S2,0.0)
            S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
            S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
            S2_state_eaares = AMAX1(S2_state_eaares,0.0)
            kflo(ifls3) = kflo(ifls3) + vol_to_inject
            flow_asr_inj(is8bsn-6,k) = vol_to_inject
            total_divers_to_eaares = vol_to_inject
     $                 + total_divers_to_eaares
          endif
         endif
        enddo
        go to 7000
6000       weircff_reg(il8bsn-6,1) = weircff(ieaa_res_asr_reg
     $                (il8bsn-6,1),1,1)
           if (opt_sim_lok_eaares .ne. 'EXCESS') then
             if (iclimate_opt .eq. 1) then
                pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
     $                            - totloketvol3(month)
     $                            - avg_demand_lok3(month)
                month_pred = month + 3
                if (month_pred .gt. 12) month_pred = month_pred - 12
                call bpts(iday,month_pred,10.0,1,n_fc_zones+3,1
     $                     ,eaares_surge_pred)
                if (lok_split_option .eq. 0) then
                  SPRES = STG2STO(rlkstg,0)
                  STO_PROJ = SPRES + pred_change_in_lok_sto
                  pred_lok_stg = STG2STO(STO_PROJ,1)
                else
                  SPRES = STG2STOLITZ(rlkstg,0)
                  STO_PROJ = SPRES + pred_change_in_lok_sto
                  pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
                endif
                if (opt_for_reg_rel_to_res(il8bsn-6,1) .eq. 'FOREC')
     $              then
                  lregtres = (pred_lok_stg .gt. eaares_surge_pred
     $              .and. itrib_hydro_cond_opt .eq. 1
     $              .and. (trib_rf_et .gt.
     $              trib_rfet_thres .or. s65e_runff_wkly
     $              .ge. s65e_runff_wkly_thres))
     $              .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
     $              mzone_puls_top .and. mzone .le. mzone_puls_bot
     $              .and. (stagelo .lt. stgref_lok_to_eaasurgt
     $              .and.climate_index(month) .gt. s_forec_eaar_thres2
     $              .or. (stagelo.ge.stgref_lok_to_eaasurgt
     $              .and. climate_index(month) .gt. s_forec_eaar_thres1
     $              )).and. izone_flex_eaares_opt .eq. 1))
     $              .and. iadd_ops_to_eaa_res_line .eq. 1)
     $              .or. (nint(S2_state_eaares) .gt. 0 .and.
```

```
     $               izone_flex_eaares_opt .eq. 0
     $               .and. iadd_ops_to_eaa_res_line .eq. 1)
     $               .or.(pred_lok_stg .gt. eaares_surge_pred
     $               .and. itrib_hydro_cond_opt .eq. 0
     $               .and.izone_flex_eaares_opt.eq.0)
                   else
                     lregtres = (nint(S2_state_eaares).gt. 0.and.
     $                   izone_flex_eaares_opt.eq.0) .or.(stagelo .gt.
     $                   stgref_lok_to_eaasurgt .and.izone_flex_eaares_opt
     $                   .eq.1)
                   endif
                   if (lregtres) then
                      vol_to_inject=amin1(volwtr(ieaa_res_asr_reg(il8bsn-6
     $                   ,1)),weircff_reg(il8bsn-6,1))
                     flow_to_eaa_res_reg(il8bsn-6,1) = vol_to_inject
     $                  + flow_to_eaa_res_reg(il8bsn-6,1)
                     volwtr(ieaa_res_asr_reg(il8bsn-6,1))= volwtr
     $                  (ieaa_res_asr_reg(il8bsn-6,1))-vol_to_inject
                     S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
                     S2_state_eaares=S2_state_eaares-vol_to_inject*1.9835
                     S2_ASR = S2_ASR - vol_to_inject*1.9835
                     S2     = S2     - vol_to_inject*1.9835
                     S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
                     S2     = AMAX1(S2,0.0)
                     S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
                     S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
                     S2_state_eaares = AMAX1(S2_state_eaares,0.0)
                     kflo(iflc10a) = kflo(iflc10a) + vol_to_inject
                     flow_asr_inj(il8bsn-6,1) = vol_to_inject
                     total_divers_to_eaares = vol_to_inject
     $                       + total_divers_to_eaares
                   else
                     flow_asr_inj(il8bsn-6,1) = 0.0
                   endif
                 else
                   avail_vol_to_res = S2_EAARRSURGT
                 endif
               else
                 IF (iasr_opt_eaa(il8bsn-6) .eq. 1 ) then
                   avail_vol_to_res = S2_EAARRSURGT
                 else
                   avail_vol_to_res = S2_state_eaares
                 endif
               endif
c
               ntarg = 0
               do itarg = 1,no_targ_loc(il8bsn-6)
                 if (LWSLO(itarg_area(itarg,il8bsn-6))) then
                   ntarg = ntarg + 1
                 endif
               enddo
c
              IF (avail_vol_to_res .GT. 0.0 .and. (iclimate_opt .eq. 0
     $           .or. opt_sim_lok_eaares .eq. 'EXCESS')) then
              IF (nint(S2) .GT. 0 .OR. (avg_lokstage .ge. prev_avg_lokstg
     $      .AND.max_days_mean.gt.0).OR.(nint(S2) .EQ. 0 .AND.
     $      ntarg .eq. 0 .and. kflo(iflc10a) .ge. 0.0)) then
```

```
              vol_to_inject=amin1(volwtr(ieaa_res_asr_reg(il8bsn-6
     $                 ,1)),weircff_reg(il8bsn-6,1))
              flow_to_eaa_res_reg(il8bsn-6,1) = vol_to_inject
     $          + flow_to_eaa_res_reg(il8bsn-6,1)
             volwtr(ieaa_res_asr_reg(il8bsn-6,1))=volwtr(ieaa_res_asr_reg
     $           (il8bsn-6,1))-vol_to_inject
              S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject*1.9835
              S2_state_eaares = S2_state_eaares - vol_to_inject*1.9835
              S2_ASR = S2_ASR - vol_to_inject*1.9835
              S2     = S2     - vol_to_inject*1.9835
              S2_REGTOEST = S2_REGTOEST - vol_to_inject*1.9835
              S2     = AMAX1(S2,0.0)
              S2_REGTOEST = AMAX1(S2_REGTOEST,0.0)
              S2_EAARRSURGT = AMAX1(S2_EAARRSURGT,0.0)
              S2_state_eaares = AMAX1(S2_state_eaares,0.0)
              kflo(iflc10a) = kflo(iflc10a) + vol_to_inject
              flow_asr_inj(il8bsn-6,1) = vol_to_inject
              total_divers_to_eaares = vol_to_inject
     $                 + total_divers_to_eaares
           endif
          endif
 7000  CONTINUE
c
c  Backflow FROM St Lucie Reservoir to LOK when Backflow from St Lucie
c   basin thru S-308 is occurring
c
        if (lok_split_option .eq. 1) then
           stglokup = stagelo_res
        else
           stglokup = rlkstg
        endif
        if ((ifix(eqs308) .lt. 0 .or. (stglokup .lt. stg_backflw_stl
     $     .and. nint(eqs308) .eq. 0)) .and. idbsnopt2 .eq. 1) then
           avail_stl_res_backflw = hres02 * ares02/1.9835
           stl_res_backflw = amin1(avail_stl_res_backflw
     $                 ,rmax_stl_res_bflw_cap)
           hres02  = hres02  - stl_res_backflw * 1.9835/ ares02
           rlkstor = rlkstor + stl_res_backflw*1.9835
        endif
c
c
C  OUTFLOW FROM LOK TO RES/ASR SYSTEMS FOR CALOOS AND ST LUCIE BASINS
C
       rlokreg2res1 = 0.0
       rlokreg2res2 = 0.0
cers
c LOK water above S2ASR line to RES/ASR
c systems in Caloos and StLucie basins
       if ((cal_asr_flag .eq. 1 .or. (cal_asr_flag .ne. 1
     $    .and. idbsnopt1 .eq. 1)) .and. ((S2 .gt. 0.0
     $    .and. iflex_clim_lokreg_to_calstlres .eq. 0) .or.
     $    ((S2 .gt. 0.0 .and. (((climate_index(month) .gt.
     $    rmin_clim_indx_thres(mzone,2,ioper_opt)).or.((trib_rf_et
     $    .gt. 5.0 .or. s65e_runff_wkly .gt. 7500.)
     $    .and. itrib_hydro_cond_opt .eq. 1)
     $    .and.mzone .ge. mzone_puls_top)
     $    .or. mzone .lt. mzone_puls_top))
```

```
     $    .and. iflex_clim_lokreg_to_calstlres .eq. 1))
     $    .and. opt_reg_lok_to_cal_res .eq. 'YES') then
        if (nint(cresbp2lok) .eq. 0) then
         avail_vol_in_res = (caloos_res_max_dpth-hres01)*ares01
         vol_to_inject = amin1(S2,avail_vol_in_res,qin_max01*1.9835)
         hres01 = hres01 + vol_to_inject / ares01
         S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject
         S2_state_eaares = S2_state_eaares - vol_to_inject
         S2_ASR = S2_ASR - vol_to_inject
         S2     = S2     - vol_to_inject
         eqs77 = eqs77 + vol_to_inject
         rlokreg2res1 = vol_to_inject/1.9835
        else
         vol_to_inject = 0.0
         rlokreg2res1  = 0.0
        endif
       endif
       if ((stl_asr_flag .eq. 1 .or. (stl_asr_flag .ne. 1
     $    .and. idbsnopt2 .eq. 1)) .and. ((S2 .gt. 0.0
     $    .and. iflex_clim_lokreg_to_calstlres .eq. 0) .or.
     $    ((S2 .gt. 0.0 .and. (((climate_index(month) .gt.
     $    rmin_clim_indx_thres(mzone,2,ioper_opt)).or.((trib_rf_et
     $    .gt. 5.0 .or. s65e_runff_wkly .gt. 7500.)
     $    .and. itrib_hydro_cond_opt .eq. 1)
     $    .and.mzone .ge. mzone_puls_top)
     $    .or. mzone .lt. mzone_puls_top))
     $    .and. iflex_clim_lokreg_to_calstlres .eq. 1))
     $    .and. opt_reg_lok_to_stl_res .eq. 'YES') then
         avail_vol_in_res = (stlucie_res_max-hres02)*ares02
         vol_to_inject = amin1(S2,avail_vol_in_res,qin_max02*1.9835)
         hres02 = hres02 + vol_to_inject / ares02
         S2_EAARRSURGT = S2_EAARRSURGT - vol_to_inject
         S2_state_eaares = S2_state_eaares - vol_to_inject
         S2_ASR = S2_ASR - vol_to_inject
         S2     = S2     - vol_to_inject
         eqs308 = eqs308 + vol_to_inject
         rlokreg2res2 = vol_to_inject/1.9835
       endif
C
C
C  REGULATORY RELEASES
C
c  compute EAA conveyance capacities for neutral case for
c  Miami and NNR canals depending on optons input as a
c  function of zones of LOK Regulation schedule
c
c
      if (mzone .le. n_fc_zones) then
       if (istaopt (is8bsn-6) .eq. 0) then
c  Miami Canal
         ireach = 1
         ntarg = 0
         do itarg = 1,no_targ_loc(is8bsn-6)
           if (LRGLO(itarg_area(itarg,is8bsn-6))) then
             ntarg = ntarg + 1
           endif
         enddo
```

```
    if (wcaenv(idn_wca_indx(is8bsn-6))
&         .eq. 'TRUE ') then
      ntarg = 0
      do itarg = 1,no_targ_loc(is8bsn-6)
       if (LWSLO(itarg_area(itarg,is8bsn-6))) then
         ntarg = ntarg + 1
       endif
      enddo
    endif
    if ((eaa_conv_opt_reg(is8bsn-6,mzone,ioper_opt) .ne. 'PUMP'
$    .and.(ntarg.eq. 0 .and.limit_reg_rel_glades(is8bsn-6,mzone,
$    ioper_opt).eq.'STGTRG')) .or.igrav_sim_opt(is8bsn-6,
$    mzone) .eq. 1) then
      if (ntarg.eq. 0 .and.limit_reg_rel_glades(is8bsn-6,mzone,
$       ioper_opt).eq.'STGTRG') ceverg_needs(is8bsn-6) = 'NO '
      call qmax_one_reach(stagelo,s8_tailw,5,1,ireach,1,
$        grf3amx,s354cap,miamic_cap_upper_grav,s8_cap,s354_tw,ctw
$        ,ifacteaa)
         s8_spill_hw  = ctw
     else
       if (eaa_conv_opt_reg(is8bsn-6,mzone,ioper_opt) .ne. 'PUMP')
$      then
        if (limit_reg_rel_glades(is8bsn-6,mzone,ioper_opt).eq.
$       'STGTRG') then
          ceverg_needs(is8bsn-6) = 'YES'
        endif
       endif
       istype = 1
       istr_eaa = 5
       chw = stagelo
       ctw = pmp_int(is8bsn-6)
       delta0 = (chw - ctw) / 25.0
       call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo
$           ,delta0,1.0,100,miamic_cap_upper_pump,s354cap,s354_tw
$           ,ifacteaa)
       if (s354_tw .gt. rmaxstg(is8bsn-6)) then
         s354_tw = rmaxstg(is8bsn-6)
         s354cap = -1.0
         call eaaspill(s354cap,stagelo,rmaxstg(is8bsn-6),5
$               ,ifacteaa)
         qcanal = -1.0
         if (s354_tw.gt.ctw) then
           call eaa_prof_lu_table(qcanal,s354_tw,ctw,
&           ireach,ifacteaa)
         else
           qcanal = 0.0
         endif
         miamic_cap_upper_pump = qcanal
        endif
        s8_cap = s8max
        grf3amx = miamic_cap_upper_pump
     endif
 else
     istype = 1
     istr_eaa = 5
     chw = stagelo
     ctw = pmp_int_new(is8bsn-6)
```

```
          delta0 = (chw - ctw) / 25.0
          ireach = 5
          call us_flow(istype,istr_eaa,ireach,1,chw,ctw,stagelo
     $           ,delta0,1.0,100,miamic_cap_upper_pump,s354cap,s354_tw
     $           ,ifacteaa)
          if (s354_tw .gt. rmaxstg(is8bsn-6)) then
            s354_tw = rmaxstg(is8bsn-6)
            s354cap = -1.0
            call eaaspill(s354cap,stagelo,rmaxstg(is8bsn-6),5,ifacteaa)
            qcanal = -1.0
            if (s354_tw.gt.ctw) then
              call eaa_prof_lu_table(qcanal,s354_tw,ctw,
     $         ireach,ifacteaa)
            else
              qcanal = 0.0
            endif
            miamic_cap_upper_pump = qcanal
           endif
           grf3amx = miamic_cap_upper_pump
           s8_cap = s8max
        endif
        s354mx = s354cap
       endif
c
c  NNR Canal
c
       if (mzone .le. n_fc_zones) then
        if (istaopt(is7bsn-6) .eq. 0) then
          ireach  = 2
          ntarg = 0
          do itarg = 1,no_targ_loc(is7bsn-6)
            if (LRGLO(itarg_area(itarg,is7bsn-6))) then
              ntarg = ntarg + 1
            endif
          enddo
          if (wcaenv(idn_wca_indx(is7bsn-6))
     &         .eq. 'TRUE ') then
            ntarg = 0
            do itarg = 1,no_targ_loc(is7bsn-6)
             if (LWSLO(itarg_area(itarg,is7bsn-6))) then
              ntarg = ntarg + 1
             endif
            enddo
          endif
          if ((eaa_conv_opt_reg(is7bsn-6,mzone,ioper_opt) .ne. 'PUMP'
     $    .and.(ntarg.eq. 0 .and.limit_reg_rel_glades(is7bsn-6,mzone,
     $    ioper_opt) .eq.'STGTRG')) .or.igrav_sim_opt(is7bsn-6,mzone)
     $    .eq. 1) then
           if (ntarg.eq. 0 .and.limit_reg_rel_glades(is7bsn-6,mzone,
     $       ioper_opt).eq.'STGTRG') ceverg_needs(is7bsn-6) = 'NO '
            call  qmax_two_reaches(stagelo,s7_tailw,6,2,ireach,2,
     $         grf2amx,s351cap,nnrc_cap_upper_grav,s7_cap,chw,ctw
     $         ,ifacteaa)
            fllks6 = grf2amx - nnrc_cap_upper_grav
            grf2amx = nnrc_cap_upper_grav
          else
            if (eaa_conv_opt_reg(is7bsn-6,mzone,ioper_opt) .ne. 'PUMP')
```

```
$      then
         if (limit_reg_rel_glades(is7bsn-6,mzone,ioper_opt).eq.
$        'STGTRG') then
           ceverg_needs(is7bsn-6) = 'YES'
         endif
       endif
       istype = 1
       istr_eaa = 6
       chw = stagelo
       ctw = pmp_int(is7bsn-6)
       delta0 = (chw - ctw) / 25.0
       call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo
$           ,delta0,1.0,100,nnrc_cap_upper_pump,s351cap
$           ,s351_tw,ifacteaa)
       if (s351_tw .gt. rmaxstg(is7bsn-6)) then
         s351_tw = rmaxstg(is7bsn-6)
         s351cap = -1.0
         call eaaspill(s351cap,stagelo,rmaxstg(is7bsn-6),6
$             ,ifacteaa)
         qcanal = -1.0
         if (s351_tw.gt.ctw) then
           call eaa_prof_lu_table(qcanal,s351_tw,ctw,
&           ireach,ifacteaa)
         else
           qcanal = 0.0
         endif
         nnrc_cap_upper_pump = qcanal
        endif
        fllks6 = -1.0
        call eaa_prof_lu_table(fllks6,s351_tw,pmp_int(is6bsn-6)
$           ,3,ifacteaa)
        s7_cap = s7max
        grf2amx = nnrc_cap_upper_pump
     endif
   else
     ireach = 6
     istype = 1
     istr_eaa = 6
     chw = stagelo
     ctw = pmp_int_new(is7bsn-6)
     delta0 = (chw - ctw) / 25.0
     call us_flow(istype,istr_eaa,ireach,2,chw,ctw,stagelo
$           ,delta0,1.0,100,nnrc_cap_upper_pump,s351cap,s351_tw
$           ,ifacteaa)
     if (s351_tw .gt. rmaxstg(is7bsn-6)) then
       s351_tw = rmaxstg(is7bsn-6)
       s351cap = -1.0
       call eaaspill(s351cap,stagelo,rmaxstg(is7bsn-6),6,ifacteaa)
       qcanal = -1.0
       if (s351_tw.gt.ctw) then
         call eaa_prof_lu_table(qcanal,s351_tw,ctw,
&           ireach,ifacteaa)
       else
         qcanal = 0.0
       endif
       nnrc_cap_upper_pump = qcanal
      endif
```

```
          fllks6 = -1.0
          call eaa_prof_lu_table(fllks6,s351_tw,pmp_int(is6bsn-6)
     $          ,3,ifacteaa)
          grf2amx = nnrc_cap_upper_pump
          s7_cap = s7max
       endif
       s351mx = s351cap
      endif
      if (mzone .le. mzone_puls_bot) then
        if (mzone .ge. mzone_puls_top) then
           clokzone_wca = 'PULSE_Z'
        else
           clokzone_wca = zoneid_p(mzone)
        endif
      else
        clokzone_wca = 'BEL_SCH'
      endif
c
     if (mzone .ge. mzone_puls_top .and. mzone .le. n_fc_zones) then
       if (itrib_hydro_cond_opt .eq. 1) then
         if (trib_rf_et .gt. trib_rf_et_thres_wca(1
     &       ,mzone) .or. s65e_runff_wkly .gt.
     &        s65e_runff_thres_wca(1,mzone)) then
           ctrib_hydro_cond_wca = cgen_trib_hydro_categ_wca(2,mzone)
         else
           ctrib_hydro_cond_wca = cgen_trib_hydro_categ_wca(1,mzone)
         endif
       endif
       if (iclimate_opt .eq. 1) then
        if ((itrib_hydro_cond_opt .eq. 1 .and. trib_rf_et
     $     .gt. trib_rf_et_thres_wca(1,mzone) .or.
     $     s65e_runff_wkly .gt. s65e_runff_thres_wca
     $     (1,mzone)) .or. multi_seas_pred_opt.eq.1) then
         if (multi_seas_pred_opt.eq.0) then
          if (climate_index(month).gt.rmin_clim_indx_thres_wca
     $       (mzone,2)) then
             cseas_forec_wca = cgen_seas_categ_wca(2,mzone)
          else
             cseas_forec_wca = cgen_seas_categ_wca(1,mzone)
          endif
         endif
         if(multi_seas_pred_opt.eq.1) then
           if(rmulti_seas_pred(month).gt.
     $          rmulti_seas_thres_wca(2,mzone)) then
             cmulti_seas_forec_wca = cgen_multi_seas_categ_wca(2,mzone)
           else
             cmulti_seas_forec_wca = cgen_multi_seas_categ_wca(1,mzone)
           endif
         endif
        endif
        if (itrib_hydro_cond_opt.eq.0) then
          if (multi_seas_pred_opt.eq.0) then
           if (climate_index(month).gt.rmin_clim_indx_thres_wca
     $       (mzone,2)) then
             cseas_forec_wca = cgen_seas_categ_wca(2,mzone)
           else
             cseas_forec_wca = cgen_seas_categ_wca(1,mzone)
```

```
            endif
           endif
         endif
        endif
      endif
c
      ntarg_hill = 0
      do itarg = 1,no_targ_loc(is6bsn-6)
         if (LRGLO(itarg_area(itarg,is6bsn-6))) then
           ntarg_hill = ntarg_hill + 1
         endif
      enddo
      ntarg_wpb = 0
      do itarg = 1,no_targ_loc(is5absn-6)
         if (LRGLO(itarg_area(itarg,is5absn-6))) then
            ntarg_wpb = ntarg_wpb + 1
         endif
      enddo
      ntarg_nnr = 0
      do itarg = 1,no_targ_loc(is7bsn-6)
         if (LRGLO(itarg_area(itarg,is7bsn-6))) then
           ntarg_nnr = ntarg_nnr + 1
         endif
      enddo
      ntarg_mia = 0
      do itarg = 1,no_targ_loc(is8bsn-6)
         if (LRGLO(itarg_area(itarg,is8bsn-6))) then
           ntarg_mia = ntarg_mia + 1
         endif
      enddo
c
      DO 600 IS = 1,NSTRCA_REG
         GOTO (200,300,400,500,550) IRGPRTY(IS)
200      IF(S2 .GT.0.0 ) THEN
C OUTLET STRUCTURE S5A DISCHARGES INTO WCA1
C RMAXSTG(IDAY,ISTOR) = STAGE IN CONSERVATION AREA ABOVE WHICH NO REGULATORY
C                     RELEASE CAN BE MADE FROM LOK TO THE CONSERVATION AREA
C ISTOR = STORAGE AREA NUMBER
C IWCA1 = CONSERVATION AREA NUMBER FOR WCA1
C
             ntarg = 0
             do itarg = 1,no_targ_loc(is5absn-6)
               if (LRGLO(itarg_area(itarg,is5absn-6))) then
                 ntarg = ntarg + 1
               endif
             enddo
             if (iclimate_opt .eq. 1 .and. mzone .ge.
     &           mzone_puls_top.and. wcaenv(idn_wca_indx(is5absn-6))
     &           .eq. 'TRUE ') then
                ntarg = 0
                do itarg = 1,no_targ_loc(is5absn-6)
                 if (LWSLO(itarg_area(itarg,is5absn-6))) then
                   ntarg = ntarg + 1
                 endif
                enddo
             endif
             rmax_des_stg = regstg_wca(idn_wca_indx(is5absn-6)
```

```
&               ,nzone(idn_wca_indx(is5absn-6)+1))
        if (trig_wca_stg(iwca1) .lt. regstg_wca(iwca1
&         ,itop_zone_indx(iwca1+1))) then
          iopt_other_wcas_reg = iopt_coth_wcas(iwca1)
        else
          iopt_other_wcas_reg = 1
        endif
        if (limit_reg_rel_glades(is5absn-6,mzone,ioper_opt) .eq.
&         'SCHED ') then
           stg_limit_reg(is5absn-6) = regstg_wca(idn_wca_indx
&               (is5absn-6),1) + offset_reg_to_wcas(is5absn-6)
           if (trig_wca_stg(iwca1) .le. stg_limit_reg
&            (is5absn-6)) then
              ceverg_needs(is5absn-6) = 'YES'
           else
              ceverg_needs(is5absn-6) = 'NO '
           endif
        elseif ((limit_reg_rel_glades(is5absn-6,mzone,ioper_opt)
&        .eq.'STGTRG' .and. mzone .ge. mzone_puls_top) .or. (mzone
&         .lt. mzone_puls_top .and. ioper_opt .eq. 2 .and.
&         rmulti_seas_pred(month) .lt. rmulti_seas_thres(2
&         ,mzone,ioper_opt) .and. rlkstg .lt. zonec_norm .and.
&         multi_seas_pred_opt.eq.1.and.iopt_drawdown .eq. 1)) then
            stg_limit_reg(is5absn-6) = -901.
            if (ntarg .gt. 0) then
              ceverg_needs(is5absn-6) = 'YES'
            else
              ceverg_needs(is5absn-6) = 'NO '
            endif
        else
          if (wcaenv(iwca1) .eq. 'FALSE') then
            stg_limit_reg(is5absn-6) = rmax_des_stg
&                 + offset_reg_to_wcas(is5absn-6)
            if (trig_wca_stg(iwca1) .le. stg_limit_reg
&            (is5absn-6)) then
              ceverg_needs(is5absn-6) = 'YES'
            else
              ceverg_needs(is5absn-6) = 'NO '
            endif
          endif
        endif
        if (opt_multi_seas_for_reg_wca .eq. 'WCA') then
          thres_multi_seas = rmulti_seas_thres_wca(2,mzone)
        else
          thres_multi_seas = rmulti_seas_thres(2,mzone,ioper_opt)
        endif
        IF ((ntarg.gt.0 .and. ntarg_nnr.gt.0 .and.
&        ntarg_mia .gt. 0 .and. multi_seas_pred_opt.eq.0
&        .and. itrib_hydro_cond_opt.eq.0) .or.
&       (ntarg .gt. 0 .and. (((rmulti_seas_pred
&       (month).gt.thres_multi_seas .and.
&       multi_seas_pred_opt.eq.1) .and. (((trib_rf_et
&       .gt. trib_rf_et_thres_wca(1,mzone) .or.
&       s65e_runff_wkly .gt.s65e_runff_thres(1,mzone))
&       .and.itrib_hydro_cond_opt.eq.1).or.
&       itrib_hydro_cond_opt.eq. 0) .and.
&       climate_index(month).gt.rmin_clim_indx_thres_wca
```

```
&      (mzone,2).and.iclimate_opt.eq.1.and.mzone.ge.mzone_puls_top)
&       .or. (mzone.lt.mzone_puls_top.and.iclimate_opt.eq.1)))
&       .or. (lok_reg_to_wca_mode .eq. 'FLDC' .or. ((trig_wca_stg
&       (iwca1) .le.stg_limit_reg(is5absn-6) .and. trig_wca_stg
&       (iwca2a) .le. stg_limit_reg(is7bsn-6) .and. trig_wca_stg
&       (iwca3a).le.stg_limit_reg(is8bsn-6).and.iopt_other_wcas_reg
&       .eq. 1).or.(trig_wca_stg(iwca1).le.stg_limit_reg
&       (is5absn-6).and. iopt_other_wcas_reg .eq. 0)
&       .and. wcaenv(iwca1).eq.'FALSE')) .or.((ntarg .gt. 0
&       .and. wcaenv(iwca1) .eq. 'TRUE ').and.iclimate_opt .eq. 0
&       .and.itrib_hydro_cond_opt.eq.0) .or. nint(s2_regtoest)
&       .gt. 0) then
          AVCAP_S352  = S352MX - KFLO(IFLHG5)
          AVCAP_S352  = AMAX1(AVCAP_S352,0.0)
          if(iconv_use_inj(is5absn-6) .eq. 0) then
            runoff_south_for_conv = runoff_south(is5absn-6)
          else
            runoff_south_for_conv
&                     = runoff_south_bef_divers(is5absn-6)
          endif
         total_lok_ws_to_sta1we=wsdel_to_sta(indx_sta1w_frm_lok_ws)
&         + wsdel_to_sta(indx_sta1e_frm_lok_ws)
&         + wsdel_to_sta(indx_sta1ee_frm_lok_ws)
          RMXRGS5A = S5AMAX*PCTWMA(IS5ABSN-6)
&                   - AMAX1(KFLO(IFLS5A)+total_lok_ws_to_sta1we,
&                  KFLO(IFLS5A5)+total_lok_ws_to_sta1we)
          RMXRGS5A = AMAX1(RMXRGS5A,0.0)
          oper_capac = cnlcap(iwpb) - (1.0-pct_des(iwpb,iseas))
$                       *cnlcap(iwpb)/rcpfacteaa(is5absn-6)
          if (.NOT. DRYSEAS .and. nint(runoff_south_for_conv)
$            .gt. 0) then
            AVAIL_REG = amin1(fllkwpb,oper_capac)
$                 - kfloref(is5absn-6)
            if (oper_capac .le. fllkwpb) then
              reg_lim_operhyd = 'OPER'
            else
              reg_lim_operhyd = 'HYDR'
            endif
          else
            AVAIL_REG = amin1(fllkwpb - kfloref(is5absn-6)
$             ,oper_capac-flow_to_import_wpb)
            if (amax1(oper_capac-flow_to_import_wpb,0.0) .lt.
$               amax1(fllkwpb - kfloref(is5absn-6),0.0)) then
              reg_lim_operhyd = 'OPER'
            else
              reg_lim_operhyd = 'HYDR'
            endif
          endif
          AVAIL_REG = amax1(AVAIL_REG,0.0)
          QS5ADL = AMIN1(AVCAP_S352,RMXRGS5A,AVAIL_REG)
          if (QS5ADL .EQ. AVCAP_S352) THEN
             reg_lim_indic(iwpb) = 'S352'
          elseif (QS5ADL .EQ. RMXRGS5A) THEN
             reg_lim_indic(iwpb) = 'S5A '
          else
             if (QS5ADL .eq. AVAIL_REG) then
               if (ifix(AVCAP_S352) .gt. 0) then
```

```
                      reg_lim_indic(iwpb) = reg_lim_operhyd
                    endif
                  endif
                endif
                QS5ADL = AMAX1(QS5ADL,0.0)
C
                REGALLOC = S2
C
                if (QS5ADL .ge. REGALLOC/1.9835) then
                   reg_lim_indic(iwpb) = 'LKSTOR'
                endif
                QS5ADL = AMIN1(QS5ADL,REGALLOC/1.9835)
                  IF (ISTAOPT(IS5ABSN-6) .EQ. 0) THEN
                   KFLO(IFLHG5) = QS5ADL + KFLO(IFLHG5)
                   REGS352 = QS5ADL
                   KFLO(IFLS5A5) = KFLO(IFLS5A5) + QS5ADL
                   KFLO(IFLS5A) = KFLO(IFLS5A) + QS5ADL
                   s5a2_so = s5a2_so + qs5adl
                   S2 = S2 - QS5ADL * 1.9835
                   S2_ASR = S2_ASR - QS5ADL * 1.9835
                   S2_EAARRSURGT = S2_EAARRSURGT - QS5ADL*1.9835
                   S2_state_eaares = S2_state_eaares - QS5ADL*1.9835
                  ELSE
                   if (REGTSTA .ge. VOLWTR(INDXSTA(IS5ABSN-6))) then
                     reg_lim_indic(iwpb) = 'STASTR'
                   endif
                   REGTSTA = AMIN1(QS5ADL,VOLWTR(INDXSTA(IS5ABSN-6)))
                   KFLO(IFLHG5) = REGTSTA + KFLO(IFLHG5)
                   REGS352 = REGTSTA
                   KFLO(IFLS5A5) = KFLO(IFLS5A5) + REGTSTA
                   KFLO(IFLS5A) = KFLO(IFLS5A) + REGTSTA
                   s5a2_so = s5a2_so + regtsta
                   VOLWTR(INDXSTA(IS5ABSN-6))=VOLWTR(INDXSTA(IS5ABSN-6))
     &                    - REGTSTA
                   S2 = S2 - REGTSTA * 1.9835
                   S2_ASR = S2_ASR - REGTSTA * 1.9835
                   S2_EAARRSURGT = S2_EAARRSURGT - REGTSTA * 1.9835
                   S2_state_eaares = S2_state_eaares - REGTSTA*1.9835
                  ENDIF
            ENDIF
         ENDIF
         GOTO 600
300      IF(S2 .GT.0.0 ) THEN
C OUTLET STRUCTURE S6 DISCHARGES INTO WCA1
C
C RMAXSTG(IDAY,ISTOR)) = STAGE IN WCA2A ABOVE WHICH NO REGULATORY RELEASE FROM
C    LAKE OKEECHOBEE WILL BE ALLOWED TO GO TO WCA2A
C TFLOWS6 = TOTAL LIMIT ON EAA RUNOFF AND WATER SUPPLY PLUS REGULATORY FLOWS
C KFLO(IFLS6) - QS6DL = EAA RUNOFF
C FLLKS6 = LIMIT ON WS AND REGULATORY RELEASES
C S6FL = LIMIT BASED ON KNOWN EAA RO & LOK WS RELEASE; MAX REGULATORY RELEASE
C    AND S6 PUMP CAPACITY
C S6MAX =  S6 PUMP CAPACITY (D/S CAPACITY)
C TFLOWS2 = MAX. POTENTIAL FLOW THROUGH S2 ASSUMING S6 WILL REACH ITS LIMIT
            ntarg = 0
            do itarg = 1,no_targ_loc(is6bsn-6)
              if (LRGLO(itarg_area(itarg,is6bsn-6))) then
```

```
            ntarg = ntarg + 1
          endif
        enddo
        if (iclimate_opt .eq. 1 .and. mzone .ge.
&            mzone_puls_top .and. wcaenv(idn_wca_indx_env(is6bsn-6))
&            .eq. 'TRUE ') then
           ntarg = 0
           do itarg = 1,no_targ_loc(is6bsn-6)
            if (LWSLO(itarg_area(itarg,is6bsn-6))) then
              ntarg = ntarg + 1
            endif
           enddo
        endif
        iwca_r = idn_wca_indx_env(is6bsn-6)
        if (trig_wca_stg(iwca_r) .lt. regstg_wca(iwca_r
&          ,itop_zone_indx(iwca_r+1))) then
           iopt_other_wcas_reg = iopt_coth_wcas(iwca_r)
        else
           iopt_other_wcas_reg = 1
        endif
        rmax_des_stg = regstg_wca(idn_wca_indx(is6bsn-6)
&                ,nzone(idn_wca_indx(is6bsn-6)+1))
        if (limit_reg_rel_glades(is6bsn-6,mzone,ioper_opt) .eq.
&         'SCHED ') then
           stg_limit_reg(is6bsn-6) = regstg_wca(idn_wca_indx
&             (is6bsn-6),1) + offset_reg_to_wcas(is6bsn-6)
           if (trig_wca_stg(iwca1) .le. stg_limit_reg
&            (is6bsn-6)) then
             ceverg_needs(is6bsn-6) = 'YES'
           else
             ceverg_needs(is6bsn-6) = 'NO '
           endif
        elseif (limit_reg_rel_glades(is6bsn-6,mzone,ioper_opt).eq.
&         'STGTRG' .and. mzone .ge. mzone_puls_top.or. (mzone
&          .lt. mzone_puls_top .and. ioper_opt .eq. 2 .and.
&          rmulti_seas_pred(month) .lt. rmulti_seas_thres(2
&          ,mzone,ioper_opt).and. rlkstg .lt. zonec_norm .and.
&          multi_seas_pred_opt.eq.1.and.iopt_drawdown .eq. 1)) then
            stg_limit_reg(is6bsn-6) = -901.
            if (ntarg .gt. 0) then
              ceverg_needs(is6bsn-6) = 'YES'
            else
              ceverg_needs(is6bsn-6) = 'NO '
            endif
        else
           if (wcaenv(iwca_r) .eq. 'FALSE') then
            stg_limit_reg(is6bsn-6) = rmax_des_stg
&             + offset_reg_to_wcas(is6bsn-6)
             if (trig_wca_stg(iwca_r) .le. stg_limit_reg
&             (is6bsn-6)) then
               ceverg_needs(is6bsn-6) = 'YES'
             else
               ceverg_needs(is6bsn-6) = 'NO '
             endif
           endif
        endif
         if (opt_multi_seas_for_reg_wca .eq. 'WCA') then
```

```
          thres_multi_seas = rmulti_seas_thres_wca(2,mzone)
        else
          thres_multi_seas = rmulti_seas_thres(2,mzone,ioper_opt)
        endif
        IF ((ntarg.gt.0 .and. ntarg_nnr.gt.0 .and.
&        ntarg_mia .gt. 0 .and. multi_seas_pred_opt.eq.0
&        .and. itrib_hydro_cond_opt.eq.0) .or.
&       (ntarg .gt. 0 .and. (((rmulti_seas_pred
&       (month).gt.thres_multi_seas .and.
&       multi_seas_pred_opt.eq.1) .and. (((trib_rf_et
&       .gt. trib_rf_et_thres_wca(1,mzone) .or.
&       s65e_runff_wkly .gt.s65e_runff_thres(1,mzone))
&       .and.itrib_hydro_cond_opt.eq.1).or.
&       itrib_hydro_cond_opt.eq. 0) .and.
&       climate_index(month).gt.rmin_clim_indx_thres_wca
&      (mzone,2).and.iclimate_opt.eq.1.and.mzone.ge.mzone_puls_top)
&       .or. (mzone.lt.mzone_puls_top.and.iclimate_opt.eq.1)))
&       .or. (lok_reg_to_wca_mode .eq. 'FLDC' .or. ((trig_wca_stg
&       (iwca1) .le.stg_limit_reg(is5absn-6) .and. trig_wca_stg
&       (iwca2a) .le. stg_limit_reg(is7bsn-6) .and. trig_wca_stg
&       (iwca3a).le.stg_limit_reg(is8bsn-6).and.iopt_other_wcas_reg
&       .eq. 1).or.(trig_wca_stg(iwca1).le.stg_limit_reg
&       (is5absn-6) .and. iopt_other_wcas_reg .eq. 0)
&       .and. wcaenv(iwca_r).eq.'FALSE')) .or.((ntarg .gt. 0
&       .and. wcaenv(iwca_r) .eq. 'TRUE ').and.iclimate_opt .eq. 0
&       .and.itrib_hydro_cond_opt.eq.0) .or. nint(s2_regtoest)
&       .gt. 0) then
         IF (KFLO(IFLS2) .GE. 0.0) THEN
          oper_capac = cnlcap(ihill) - (1.0-pct_des(ihill,iseas))
$                         *cnlcap(ihill)/rcpfacteaa(is6bsn-6)
          if (iconv_use_inj(is6bsn-6) .eq. 0) then
             runoff_south_for_conv = runoff_south(is6bsn-6)+divers
          else
             runoff_south_for_conv
&                     = runoff_south_bef_divers(is6bsn-6)
          endif
          if (.NOT. DRYSEAS .and. nint(runoff_south_for_conv)
$            .gt. 0) then
            fllks6_reg = amin1(fllks6,oper_capac)-kfloref(is6bsn-6)
            if (oper_capac .le. fllks6) then
              reg_lim_operhyd = 'OPER'
            else
              reg_lim_operhyd = 'HYDR'
            endif
          else
            fllks6_reg = amin1(fllks6-kfloref(is6bsn-6)
$                  ,oper_capac-flow_to_import_hill
$                  -wsdel_to_sta(indxsta(is6bsn-6)))
            rlim1 = amax1(fllks6-kfloref(is6bsn-6),0.0)
            rlim2 = amax1(oper_capac-flow_to_import_hill
$             -wsdel_to_sta(indxsta(is6bsn-6)),0.0)
            if (rlim2 .lt. rlim1) then
              reg_lim_operhyd = 'OPER'
            else
              reg_lim_operhyd = 'HYDR'
            endif
          endif
```

```
                fllks6_reg = amax1(fllks6_reg,0.0)
                TFLOWS6 = KFLO(IFLS6) + fllks6_reg
                if (TFLOWS6 .ge. S6MAX) then
                  reg_lim_indic(ihill) = 'S6 '
                endif

                S6FL = AMIN1(TFLOWS6,S6MAX)
                TFLOWS2 = KFLO(IFLS2) + S6FL - KFLO(IFLS6)
                if (TFLOWS2 .ge. S351MX) then
                  reg_lim_indic(ihill) = 'S351'
                endif
                if (TFLOWS6 .lt. S6MAX .and. TFLOWS2 .lt. S351MX) then
                  reg_lim_indic(ihill) = reg_lim_operhyd
                endif
C S351MX = HG4 CAPACITY (U/S CAPACITY)
C EXCDCAP = REDUCTION IN FLOW CAPACITY ALONG NNR CANAL JUST IN CASE
C    S2 WILL REACH ITS LIMIT BEFORE S6 DOES
                EXCDCAP = AMAX1(TFLOWS2 - S351MX,0.0)
                REGFLW = S6FL - KFLO(IFLS6) - EXCDCAP
c                 print *,REGFLW,KFLO(IFLS6),EXCDCAP,TFLOWS2,S351MX
c    $                ,MONTH,IDAY,IFYR+IYEAR-1,fllks6,
c    $                runoff_south_for_conv,' S6REG  '
c    $                ,trig_wca_stg(iwca1),stg_limit_reg(is6bsn-6)
c    $                ,water_from_lok_s6
C
                  REGALLOC = S2
C
                  if (REGFLW .ge. REGALLOC/1.9835) then
                    reg_lim_indic(ihill) = 'LKSTOR'
                  endif
                  REGFLW = AMIN1(REGFLW,REGALLOC/1.9835)
                  IF (ISTAOPT(IS6BSN-6) .EQ. 0) THEN
                      KFLO(IFLS2) = KFLO(IFLS2) + REGFLW
                      REGS351 = REGS351 + REGFLW
                      KFLO(IFLS6) = REGFLW + KFLO(IFLS6)
                      S2 = S2 - REGFLW * 1.9835
                      S2_ASR = S2_ASR - REGFLW * 1.9835
                      S2_EAARRSURGT = S2_EAARRSURGT - REGFLW * 1.9835
                      S2_state_eaares = S2_state_eaares - REGFLW * 1.9835
                      hlsbreg = REGFLW
                  ELSE
                     REGTSTA = AMIN1(REGFLW,VOLWTR(INDXSTA(IS6BSN-6)),
     &                    S6FL - QINFLW(INDXSTA(IS6BSN-6),1))
                     if (REGTSTA .eq. VOLWTR(INDXSTA(IS6BSN-6))) then
                       reg_lim_indic(ihill) = 'STASTR'
                     endif
                     KFLO(IFLS2) = REGTSTA + KFLO(IFLS2)
                     REGS351 = REGS351 + REGTSTA
                     hlsbreg = REGTSTA
                     QINFLW(INDXSTA(IS6BSN-6),1)=QINFLW(INDXSTA(IS6BSN-6)
     &              ,1)  + REGTSTA
                     VOLWTR(INDXSTA(IS6BSN-6))=VOLWTR(INDXSTA(IS6BSN-6))
     &                    - REGTSTA
                     KFLO(IFLS6) = KFLO(IFLS6) + REGTSTA
                     S2 = S2 - REGTSTA * 1.9835
                     S2_ASR = S2_ASR - REGTSTA * 1.9835
                     S2_EAARRSURGT = S2_EAARRSURGT - REGTSTA * 1.9835
```

```
                  S2_state_eaares = S2_state_eaares - REGTSTA * 1.9835
                ENDIF
              else
                reg_lim_indic(ihill) = 'FLCBKP'
              endif
            endif
          ENDIF
          GOTO 600
400       IF(S2 .GT.0.0) THEN
C OUTLET STRUCTURE S7 DISCHARGES INTO WCA2A
            ntarg = 0
            do itarg = 1,no_targ_loc(is7bsn-6)
              if (LRGLO(itarg_area(itarg,is7bsn-6))) then
                ntarg = ntarg + 1
              endif
            enddo
            if (iclimate_opt .eq. 1 .and. mzone .ge.
     &            mzone_puls_top .and. wcaenv(idn_wca_indx_env(is7bsn-6))
     &            .eq. 'TRUE ') then
               ntarg = 0
               do itarg = 1,no_targ_loc(is7bsn-6)
                if (LWSLO(itarg_area(itarg,is7bsn-6))) then
                  ntarg = ntarg + 1
                endif
               enddo
            endif
            iwca_r = idn_wca_indx(is7bsn-6)
            if (trig_wca_stg(iwca_r) .lt. regstg_wca(iwca_r
     &         ,itop_zone_indx(iwca_r+1))) then
               iopt_other_wcas_reg = iopt_coth_wcas(iwca_r)
            else
               iopt_other_wcas_reg = 1
            endif
            rmax_des_stg = regstg_wca(idn_wca_indx(is7bsn-6)
     &                ,nzone(idn_wca_indx(is7bsn-6)+1))
            if (limit_reg_rel_glades(is7bsn-6,mzone,ioper_opt) .eq.
     &         'SCHED ') then
              stg_limit_reg(is7bsn-6) = regstg_wca(idn_wca_indx
     &           (is7bsn-6),2) + offset_reg_to_wcas(is7bsn-6)
              if ( wcaenv(iwca_r).eq.'FALSE') then
                if (trig_wca_stg(iwca_r) .le. stg_limit_reg
     &            (is7bsn-6)) then
                  ceverg_needs(is7bsn-6) = 'YES'
                else
                   ceverg_needs(is7bsn-6) = 'NO '
                endif
              else
                if (ntarg .gt. 0) then
                  ceverg_needs(is7bsn-6) = 'YES'
                else
                  ceverg_needs(is7bsn-6) = 'NO '
                endif
              endif
            elseif (limit_reg_rel_glades(is7bsn-6,mzone,ioper_opt).eq.
     &         'STGTRG' .and. mzone .ge. mzone_puls_top.or. (mzone
     &          .lt. mzone_puls_top .and. ioper_opt .eq. 2 .and.
     &          rmulti_seas_pred(month) .lt. rmulti_seas_thres(2
```

```
&         ,mzone,ioper_opt) .and. rlkstg .lt. zonec_norm .and.
&         multi_seas_pred_opt.eq.1.and.iopt_drawdown .eq. 1)) then
          stg_limit_reg(is7bsn-6) = -901.
          if (ntarg .gt. 0) then
            ceverg_needs(is7bsn-6) = 'YES'
          else
            ceverg_needs(is7bsn-6) = 'NO '
          endif
      else
          stg_limit_reg(is7bsn-6) = rmax_des_stg
&             + offset_reg_to_wcas(is7bsn-6)
         if ( wcaenv(iwca_r).eq.'FALSE') then
          if (trig_wca_stg(iwca_r) .le. stg_limit_reg
&           (is7bsn-6)) then
             ceverg_needs(is7bsn-6) = 'YES'
          else
             ceverg_needs(is7bsn-6) = 'NO '
          endif
         else
          if (ntarg .gt. 0) then
            ceverg_needs(is7bsn-6) = 'YES'
          else
            ceverg_needs(is7bsn-6) = 'NO '
          endif
         endif
      endif
      if (opt_multi_seas_for_reg_wca .eq. 'WCA') then
        thres_multi_seas = rmulti_seas_thres_wca(2,mzone)
      else
        thres_multi_seas = rmulti_seas_thres(2,mzone,ioper_opt)
      endif
      IF ((ntarg.gt.0 .and.
&        ntarg_mia .gt. 0 .and. multi_seas_pred_opt.eq.0
&        .and. itrib_hydro_cond_opt.eq.0) .or.
&       (ntarg .gt. 0 .and. (((rmulti_seas_pred
&       (month).gt.thres_multi_seas .and.
&       multi_seas_pred_opt.eq.1) .and. (((trib_rf_et
&       .gt. trib_rf_et_thres_wca(1,mzone) .or.
&       s65e_runff_wkly .gt.s65e_runff_thres(1,mzone))
&       .and.itrib_hydro_cond_opt.eq.1).or.
&       itrib_hydro_cond_opt.eq. 0) .and.
&       climate_index(month).gt.rmin_clim_indx_thres_wca
&      (mzone,2).and.iclimate_opt.eq.1.and.mzone.ge.mzone_puls_top)
&       .or. (mzone.lt.mzone_puls_top.and.iclimate_opt.eq.1)))
&       .or. (lok_reg_to_wca_mode .eq. 'FLDC' .or. ((trig_wca_stg
&       (iwca_r) .le. stg_limit_reg(is7bsn-6) .and. trig_wca_stg
&       (iwca3a).le.stg_limit_reg(is8bsn-6).and.iopt_other_wcas_reg
&       .eq. 1).or.(trig_wca_stg(iwca_r) .le. stg_limit_reg
&       (is7bsn-6) .and. iopt_other_wcas_reg .eq. 0) .and.
&       wcaenv(iwca_r).eq.'FALSE')) .or.((ntarg .gt. 0
&       .and. wcaenv(iwca_r) .eq. 'TRUE ').and.iclimate_opt .eq. 0
&       .and.itrib_hydro_cond_opt.eq.0) .or. nint(s2_regtoest)
&       .gt. 0) then
       IF(KFLO(IFLS2) .GE. 0.0 ) THEN
          oper_capac = cnlcap(innr) - (1.0-pct_des(innr,iseas))
$                    *cnlcap(innr)/rcpfacteaa(is7bsn-6)
          if (iconv_use_inj(is7bsn-6) .eq. 0) then
```

```
                  conv_red_runoff = runoff_south(is7bsn-6)
                else
                  conv_red_runoff = runoff_south_bef_divers(is7bsn-6)
                endif
                  if (.NOT. DRYSEAS .and. nint(conv_red_runoff)
     $                .gt. 0) then
                    reg_flow = amin1(grf2amx,oper_capac) - kfloref
     $                 (is7bsn-6)
                    reg_flow = amin1(reg_flow,s7_cap - kflo(ifls7))
                    if (oper_capac .le. grf2amx) then
                      reg_lim_operhyd = 'OPER'
                    else
                      reg_lim_operhyd = 'HYDR'
                    endif
                  else
                    reg_flow = amin1(grf2amx-kfloref(is7bsn-6)
     $                         ,oper_capac-flow_to_import_nnrc
     $                         ,s7_cap - kflo(ifls7))
                    reg1 = amax1(grf2amx-kfloref(is7bsn-6),0.0)
                    reg2 = amax1(oper_capac-flow_to_import_nnrc,0.0)
                    if (reg2 .lt. reg1) then
                      reg_lim_operhyd = 'OPER'
                    else
                      reg_lim_operhyd = 'HYDR'
                    endif
                  endif
                  reg_flow = amin1(reg_flow,s351mx-kflo(ifls2))
                  reg_flow = amax1(reg_flow,0.0)
                  if (reg_flow .ge. s351mx-kflo(ifls2)) then
                    reg_lim_indic(innr) = 'S351 '
                  else
                    reg_lim_indic(innr) = reg_lim_operhyd
                  endif
c                 print *,reg_flow,flow_to_import_nnrc,nnrctwca3a
c    $                      ,lwslo(iwca3a),wslk2a,dmnd_nnr,kfloref
c    $                      ,oper_capac,RMKVOLSTA4,grf2amx,s351mx
c    $                      ,kflo(ifls2),'  S7REG'
C
                  REGALLOC = S2
C
                if (reg_flow .ge. REGALLOC/1.9835) then
                  reg_lim_indic(innr) =  'LKSTOR'
                endif
                reg_flow_s7 = AMIN1(reg_flow,REGALLOC/1.9835)
                reg_flow_s7 = amax1(reg_flow_s7,0.0)
                   IF (ISTAOPT(IS7BSN-6) .EQ. 0) THEN
                     KFLO(IFLS2) = reg_flow_s7 + KFLO(IFLS2)
                     REGS351 = REGS351 + reg_flow_s7
C
                     KFLO(IFLS7)= KFLO(IFLS7)+reg_flow_s7
                     S2 = S2 - reg_flow_s7 * 1.9835
                     S2_ASR = S2_ASR - reg_flow_s7 * 1.9835
                     S2_EAARRSURGT = S2_EAARRSURGT - reg_flow_s7 * 1.9835
                     S2_state_eaares=S2_state_eaares-reg_flow_s7*1.9835
                     nnrreg = reg_flow_s7
                     reg_flow_to_wca2 = reg_flow_s7
                    if (eaa_conv_opt_reg(is7bsn-6,mzone,ioper_opt) .ne.
```

```
     $                  'PUMP')  then
                       s7_grav_flow = s7_grav_flow + reg_flow_s7
                    else
                       s7_pump_flow = s7_pump_flow + reg_flow_s7
                    endif
                   ELSE
                     if (reg_flow_s7 .ge. VOLWTR(INDXSTA(IS7BSN-6))) then
                        reg_lim_indic(innr) = 'STASTR'
                     endif
                     REGTSTA=AMIN1(reg_flow_s7,VOLWTR(INDXSTA(IS7BSN-6))
     &                       ,g370cap-QINFLW(INDXSTA(IS7BSN-6),2))
                     KFLO(IFLS2)= REGTSTA + KFLO(IFLS2)
                     REGS351 = REGS351 + REGTSTA
                     nnrreg = REGTSTA
                     QINFLW(INDXSTA(IS7BSN-6),2)=QINFLW(INDXSTA(IS7BSN-6)
     &                      ,2) + REGTSTA
                     volwtr(INDXSTA(IS7BSN-6)) =volwtr(INDXSTA(IS7BSN-6))
     &                      - REGTSTA
                     S2 = S2 - REGTSTA * 1.9835
                     S2_ASR = S2_ASR - REGTSTA * 1.9835
                     S2_EAARRSURGT = S2_EAARRSURGT - REGTSTA * 1.9835
                     S2_state_eaares=S2_state_eaares-REGTSTA * 1.9835
                     reg_flow_to_wca2 = 0.0
                   ENDIF
            ELSE
              reg_lim_indic(innr) = 'FLCBKP'
            ENDIF
          ENDIF
        ENDIF
        GOTO 600
500     IF(S2 .GT.0.0) THEN
          ntarg = 0
          do itarg = 1,no_targ_loc(is8bsn-6)
            if (LRGLO(itarg_area(itarg,is8bsn-6))) then
              ntarg = ntarg + 1
            endif
          enddo
          if (iclimate_opt .eq. 1 .and. mzone .ge.
     &          mzone_puls_top .and. wcaenv(idn_wca_indx(is8bsn-6))
     &          .eq. 'TRUE ') then
            ntarg = 0
            do itarg = 1,no_targ_loc(is8bsn-6)
             if (LWSLO(itarg_area(itarg,is8bsn-6))) then
               ntarg = ntarg + 1
             endif
            enddo
          endif
          rmax_des_stg = regstg_wca(idn_wca_indx(is8bsn-6)
     &                ,nzone(idn_wca_indx(is8bsn-6)+1))
          if (limit_reg_rel_glades(is8bsn-6,mzone,ioper_opt) .eq.
     $       'SCHED ') then
             stg_limit_reg(is8bsn-6) = regstg_wca(idn_wca_indx
     $          (is8bsn-6),2) + offset_reg_to_wcas(is8bsn-6)
            if ( wcaenv(iwca3a).eq.'FALSE') then
             if (trig_wca_stg(iwca3a) .le. stg_limit_reg
     &          (is8bsn-6)) then
              ceverg_needs(is8bsn-6) = 'YES'
```

```
            else
               ceverg_needs(is8bsn-6) = 'NO '
            endif
           else
            if (ntarg .gt. 0) then
              ceverg_needs(is8bsn-6) = 'YES'
            else
              ceverg_needs(is8bsn-6) = 'NO '
            endif
           endif
        elseif (limit_reg_rel_glades(is8bsn-6,mzone,ioper_opt) .eq.
&          'STGTRG' .and. mzone .ge. mzone_puls_top.or. (mzone
&           .lt. mzone_puls_top .and. ioper_opt .eq. 2 .and.
&           rmulti_seas_pred(month) .lt. rmulti_seas_thres(2
&           ,mzone,ioper_opt).and. rlkstg .lt. zonec_norm .and.
&           multi_seas_pred_opt.eq.1.and.iopt_drawdown .eq. 1)) then
            stg_limit_reg(is8bsn-6) = -901.
            if (ntarg .gt. 0) then
              ceverg_needs(is8bsn-6) = 'YES'
            else
              ceverg_needs(is8bsn-6) = 'NO '
            endif
        else
            stg_limit_reg(is8bsn-6) = rmax_des_stg
&              + offset_reg_to_wcas(is8bsn-6)
           if ( wcaenv(iwca3a).eq.'FALSE') then
            if (trig_wca_stg(iwca3a) .le. stg_limit_reg
&             (is8bsn-6)) then
               ceverg_needs(is8bsn-6) = 'YES'
            else
               ceverg_needs(is8bsn-6) = 'NO '
            endif
           else
            if (ntarg .gt. 0) then
              ceverg_needs(is8bsn-6) = 'YES'
            else
              ceverg_needs(is8bsn-6) = 'NO '
            endif
           endif
        endif
        IF ((ntarg.gt.0 .and.  multi_seas_pred_opt.eq.0
&          .and. itrib_hydro_cond_opt.eq.0) .or.
&         (ntarg .gt. 0 .and. (((rmulti_seas_pred
&         (month).gt.thres_multi_seas .and.
&         multi_seas_pred_opt.eq.1) .and. (((trib_rf_et
&         .gt. trib_rf_et_thres_wca(1,mzone) .or.
&         s65e_runff_wkly .gt.s65e_runff_thres(1,mzone))
&         .and.itrib_hydro_cond_opt.eq.1).or.
&         itrib_hydro_cond_opt.eq. 0) .and.
&         climate_index(month).gt.rmin_clim_indx_thres_wca
&        (mzone,2).and.iclimate_opt.eq.1.and.mzone.ge.mzone_puls_top)
&         .or. (mzone.lt.mzone_puls_top.and.iclimate_opt.eq.1)))
&         .or. (lok_reg_to_wca_mode .eq. 'FLDC' .or. (trig_wca_stg
&         (iwca3a) .le. stg_limit_reg(is8bsn-6) .and.
&         wcaenv(iwca3a).eq.'FALSE')) .or.((ntarg .gt. 0
&         .and. wcaenv(iwca3a) .eq. 'TRUE ').and.iclimate_opt .eq. 0
&         .and.itrib_hydro_cond_opt.eq.0) .or. nint(s2_regtoest)
```

```
     &       .gt. 0) then
C OUTLET STRUCTURE S8 DISCHARGES INTO WCA3A
             IF(KFLO(IFLS3) .GE. 0.0) THEN
                   oper_capac=cnlcap(imiami) -(1.0-pct_des(imiami,iseas))
     $                     *cnlcap(imiami)/rcpfacteaa(is8bsn-6)
                   rem_cap_s354 = s354mx - kflo(ifls3)
                   if (iconv_use_inj(is8bsn-6) .eq. 0) then
                     conv_red_runoff = runoff_south(is8bsn-6)
                   else
                     conv_red_runoff = runoff_south_bef_divers(is8bsn-6)
                   endif
                   if (.NOT. DRYSEAS .and. nint(conv_red_runoff)
     $                .gt. 0) then
                      reg_flow = amin1(grf3amx,oper_capac)
     $                                - kfloref(is8bsn-6)
                      reg_flow = amin1(reg_flow
     $                                ,s8_cap-kflo(ifls8))
                      if (oper_capac .le. grf3amx) then
                        reg_lim_operhyd = 'OPER'
                      else
                        reg_lim_operhyd = 'HYDR'
                      endif
                      reg_flow = amin1(reg_flow,rem_cap_s354)
                   else
                      reg1 = amax1(grf3amx - kfloref(is8bsn-6),0.0)
                      reg2 = amax1(oper_capac-flow_to_import_miami,0.0)
                      if (reg2 .lt. reg1) then
                        reg_lim_operhyd = 'OPER'
                      else
                        reg_lim_operhyd = 'HYDR'
                      endif
                      reg_flow = amin1(grf3amx - kfloref(is8bsn-6),
     $                    oper_capac -flow_to_import_miami,rem_cap_s354
     $                    ,s8_cap-kflo(ifls8))
                   endif
                   reg_flow = amax1(reg_flow,0.0)
                   if (reg_flow .ge. rem_cap_s354) then
                     reg_lim_indic(imiami) = 'S354 '
                   else
                     reg_lim_indic(imiami) = reg_lim_operhyd
                   endif
c                  print *,LRGLO(iwca3a),ntarg,reg_flow,grf3amx,kfloref
c    $            ,flow_to_import_miami,rem_cap_s354,stg_limit_reg(is8bsn
c    $           -6),s354mx,kflo(ifls3),' S8REG ',oper_capac,s8_cap
c    $            ,kflo(ifls8),multi_seas_pred_opt,itrib_hydro_cond_opt
c    $            ,limit_reg_rel_glades(is8bsn-6,mzone,ioper_opt),mzone
c    $            ,mzone_puls_top,climate_index(month)
C
C
                  REGALLOC = S2
C
                   if (reg_flow .ge. REGALLOC/1.9835) then
                     reg_lim_indic(imiami) =  'LKSTOR'
                   endif
                   reg_flow = AMIN1(reg_flow,REGALLOC/1.9835)
                   IF (ISTAOPT(IS8BSN-6) .EQ. 0) THEN
                    KFLO(IFLS3) = KFLO(IFLS3) + reg_flow
```

```
                    KFLO(IFLS8) = KFLO(IFLS8) + reg_flow
                    REGS354 = reg_flow
                    if (eaa_conv_opt_reg(is8bsn-6,mzone,ioper_opt) .ne.
     $                   'PUMP')  then
                       s8_grav_flow = s8_grav_flow + reg_flow
                    else
                       s8_pump_flow = s8_pump_flow + reg_flow
                    endif
C
                    S2 = S2 - reg_flow * 1.9835
                    S2_ASR = S2_ASR - reg_flow * 1.9835
                    S2_EAARRSURGT = S2_EAARRSURGT - reg_flow * 1.9835
                    S2_state_eaares = S2_state_eaares - reg_flow * 1.9835
                   ELSE
                    if (reg_flow .ge. VOLWTR(INDXSTA(IS8BSN-6))) then
                      reg_lim_indic(imiami) = 'STASTR'
                    endif
                    REGTSTA = AMIN1(reg_flow,VOLWTR(INDXSTA(IS8BSN-6))
     &                     ,g372cap-QINFLW(INDXSTA(IS8BSN-6),IS8BSN-6)
     &                     - env_ws_from_res_thru_g372)
c                   print *,REGTSTA,reg_flow,g372cap
c    &              ,QINFLW(INDXSTA(IS8BSN-6),IS8BSN-6)
c    &              ,VOLWTR(INDXSTA(IS8BSN-6)),grf3amx,kfloref(is8bsn-6)
c    &              ,s8_cap,kflo(ifls8)
                    KFLO(IFLS3) = REGTSTA + KFLO(IFLS3)
                    REGS354 = REGTSTA
                   QINFLW(INDXSTA(IS8BSN-6),IS8BSN-6) = QINFLW(INDXSTA
     &               (IS8BSN-6),IS8BSN-6) + REGTSTA
                   volwtr(INDXSTA(IS8BSN-6)) = volwtr(INDXSTA(IS8BSN-6))
     &                 - REGTSTA
                   S2 = S2 - REGTSTA*1.9835
                   S2_ASR = S2_ASR - REGTSTA*1.9835
                   S2_EAARRSURGT = S2_EAARRSURGT - REGTSTA*1.9835
                   S2_state_eaares = S2_state_eaares - REGTSTA*1.9835
                   ENDIF
             ELSE
               reg_lim_indic(imiami) = 'FLCBKP'
             ENDIF
           ENDIF
         ENDIF
         GO TO 600
 550     IF(S2 .GT.0.0 ) THEN
             IF(nint(KFLO(IFLC10A)) .GE. 0) THEN
               IF (.not.opt_l8_prop) THEN
                 IF (stagelo.gt.chdep(il8)+hdc(il8,iseas)) THEN
                   hdiff=stagelo-chdep(il8)-hdc(il8,iseas)
                 ELSE
                   hdiff=chdep(il8)+hdc(il8,iseas)-stagelo
                 END IF
C
                 capl8=cnlcap(il8bsn-6)
C
                 capac=capacc10a_bak*sqrt(hdiff)
                 avail_cap_reg = max(capac - KFLO(IFLC10A),0.0)
                 avail_canl_cap = max(capl8 - KFLO(IFLC10A),0.0)
                 rem_s5ae_cap = max(s5a4max - kflo(iflbel5),0.0)
                 reg_flow = min(avail_cap_reg,avail_canl_cap,
```

```
     &          rem_s5ae_cap,s2/1.9835)
c               print *,reg_flow,KFLO(IFLC10A),capac,avail_cap_reg
c    &           ,avail_canl_cap,cnlcap(il8bsn-6),rem_s5ae_cap
c    &           ,'  LKRGL8'
                rlokreg_to_l8 = reg_flow
                kflo(iflc10a) = kflo(iflc10a) + reg_flow
                kflo(ifll8) = kflo(ifll8) + reg_flow
                s5a3_so = s5a3_so + reg_flow
                kflo(iflbel5) = kflo(iflbel5) + reg_flow
              ENDIF
            ELSE
              reg_flow = 0.0
              rlokreg_to_l8 = 0.0
            ENDIF
          ENDIF
600   CONTINUE
c
C
C TRANSFER CODE FOR ROUTING REGULATORY RELEASES TO STA'S FOR TREATMENT
C HERE !!!
C
C<<
C
C--------------
C   ROUTE APPROPRIATE PORTION OF RUNOFF INTO
C   STAS
C--------------
C
         IF (NMAREA .GT. 0) THEN
          DO  IA = 1,NSTARFF
           IRFSTA = ISTARF(IA)
           DO   J = 1,NINSTR(IRFSTA)
            DO   K = 1,NINDPTS(IRFSTA,J)
             IF (INODOPT(IRFSTA,J,1) .EQ. 1) THEN
               IF (sfactor(IRFSTA) .GE. sfactmin .OR. ires_small_sim
     $         (IRFSTA) .eq. 'NO ') THEN
                  IF (resname(IRFSTA) .ne.name_res_spec_code(2)) THEN
                    POND(IUPSTRM(IRFSTA,J,K)) =POND(IUPSTRM(IRFSTA,J,K))
     $              + QINFLW(IRFSTA,J)*86400./(NINDPTS(IRFSTA,J)*GDAR
     $              * (1.0-carfact(IUPSTRM(IRFSTA,J,K))))
                  ENDIF
               ELSE
                  STR_INFLOW = QINFLW(IRFSTA,J)/NINDPTS(IRFSTA,J)
                  do kk = 1,noresincell(iupstrm(IRFSTA,J,K))
                     if(IRFSTA .EQ. ires_index_in_cell
     $               (iupstrm(IRFSTA,J,K),kk)) THEN
                       depth_to_wt = (ells_in_res(iupstrm
     $                 (IRFSTA,J,K),kk) - stage_res(iupstrm
     $                 (IRFSTA,J,K),kk))*s(iupstrm
     $                 (IRFSTA,J,K))
                       depth_to_wt = amax1(depth_to_wt,0.0)
                       stage_res(iupstrm(IRFSTA,J,K),kk)
     $                 =stage_res(iupstrm(IRFSTA,J,K),kk)
     $                 +amin1(STR_INFLOW*86400./(sfactor(IRFSTA)
     $            *gdar),depth_to_wt)/s(iupstrm(IRFSTA,J,K))
     $                 + amax1(STR_INFLOW*86400./(sfactor(IRFSTA)
     $            *gdar) - depth_to_wt,0.0)
```

```
                      pond_in_res(iupstrm(IRFSTA,J,K),kk)
     $                = amax1(stage_res(iupstrm(IRFSTA,J,K),kk)
     $                - ells_in_res(iupstrm(IRFSTA,J,K),kk)
     $                , 0.0)
                    endif
                  enddo
                  res_struc_inflw(IRFSTA) = res_struc_inflw(IRFSTA)
     $            + QINFLW(IRFSTA,J)*1.9835
               ENDIF
             ELSE
               QU(INCNL(IRFSTA,J,K)) = QU(INCNL(IRFSTA,J,K))
     $                  + QINFLW(IRFSTA,J) * 86400.
             ENDIF
            ENDDO
            IF (resname(IRFSTA) .ne.name_res_spec_code(2)) THEN
              QSTAIN(IRFSTA) = QSTAIN(IRFSTA) + QINFLW(IRFSTA,J)
            ENDIF
           ENDDO
          ENDDO
         ENDIF
C NOTES: * VOLUME OF STORAGE ABOVE REGULATION FROM THIS SUBROUTINE WILL BE
C        A LIMIT FOR REGULATORY RELEASES THROUGH THE CALOOSAHATCHEE AND
C        ST. LUCIE CANALS
C        $ STORAGE MUST BE CONVERTED BACK TO STAGE: STAGELO = FN(S**)
         RLKSTOR = RLKSTOR + S2_ASR
C
      DO 75 I = 1,NCALCPT
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS)=KFLO( ICALCPT(I) )
75    CONTINUE
c
c     inject LOK water into proposed ASR system when stage
c     in LOK is sufficiently high and retrieve water from ASR
c     back into LOK when LOK is sufficiently low
c
      if (iclimate_opt .eq. 0) then
        change_lok_stg = amax1(rlkstg - rstgpls(iplsday),0.0)
        if ((change_lok_stg .gt. 0.0 .and. nint(s2_asr) .gt.
     $      0 .and. nint(S2) .eq. 0 ) .or. nint(S2) .gt. 0) then
          vol_in = s2_asr
        else
          vol_in = 0.0
        endif
      else
        pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
     $                           - totloketvol3(month)
     $                           - avg_demand_lok3(month)
        month_pred = month + 3
        if (month_pred .gt. 12) month_pred = month_pred - 12
        call bpts(iday,month_pred,10.0,1,n_fc_zones+1,1,stgref_asr_pred)
        if (lok_split_option .eq. 0) then
          SPRES = STG2STO(rlkstg,0)
          STO_PROJ = SPRES + pred_change_in_lok_sto
          pred_lok_stg = STG2STO(STO_PROJ,1)
        else
          SPRES = STG2STOLITZ(rlkstg,0)
          STO_PROJ = SPRES + pred_change_in_lok_sto
```

```
          pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
        endif
        if (pred_lok_stg .ge. stgref_asr_pred) then
           vol_in = 999999.
        else
           vol_in = 0.0
        endif
      endif
      DO    I = 1,nasr_to_lok(1)
        call asr(asrnum(4,1,I,1),1,vol_in,vol_not_inject,0
     $     ,tot_cnl_dmd_asr)
        RLKSTOR = RLKSTOR - (vol_in - vol_not_inject)
        lok2asr = (vol_in - vol_not_inject)/1.9835
        s2_asr = s2_asr - lok2asr*1.9835
        vol_in = vol_in - lok2asr*1.9835
      ENDDO
c
c     recovery to LOK
c
      if (iclimate_opt .eq. 0) then
        if (rlkstg .lt. stgref_asr_rec) then
          vol_in = 9999.
        else
          vol_in = 0.0
        endif
      else
        if (DRYSEAS) then
          if (iday .eq. numd(month)) nmonth_dry = nmonth_dry + 1
          addit_demand = 200000. + (7 - nmonth_dry)*600000./7
          pred_change_in_lok_sto = climate_index(month)*1.0E6
     $             - totloketvol6(month) - addit_demand
          month_pred = month + 6
          if (month_pred .gt. 12) month_pred = month_pred - 12
          call bpts(iday,month_pred,10.0,1,n_fc_zones+4,1
     $        ,stgref_asr_rec_pred)
          if (lok_split_option .eq. 0) then
            SPRES = STG2STO(rlkstg,0)
            STO_PROJ = SPRES + pred_change_in_lok_sto
            pred_lok_stg = STG2STO(STO_PROJ,1)
          else
            SPRES = STG2STOLITZ(rlkstg,0)
            STO_PROJ = SPRES + pred_change_in_lok_sto
            pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
          endif
          if (pred_lok_stg .le. stgref_asr_rec_pred
     $        .or. rlkstg .lt. stgref_asr_rec) then
             vol_in = 9999.
          else
             vol_in = 0.0
          endif
        else
          if (rlkstg .lt. stgref_asr_rec .and. climate_index(month)
     $            .lt. 1.5) then
            vol_in = 9999.
          else
            vol_in = 0.0
          endif
```

```
          nmonth_dry = 0
        endif
      endif
      DO    I = 1,nasr_from_lok(1)
        call asr(asrnum(4,1,I,2),2,vol_in,bal_vol,1
     $     ,tot_cnl_dmd_asr)
        RLKSTOR = RLKSTOR + (vol_in - bal_vol)
        asr2lok = (vol_in - bal_vol)/1.9835
      ENDDO
c
c  compute volume of water to be routed to north storage reservoir
c   from LOK or taken from North storage to LOK depending on
c   hydrologic conditions
c
      if (inorth_stor_opt .eq. 1) then
       if (day1) hres0 = 0.0
       if (iclimate_opt .eq. 0) then
         change_lok_stg = amax1(rlkstg - rstgpls(iplsday),0.0)
         if (no_puls_zones .gt. 0) then
            ref_stage_lok = stgref_pulse_bot
         else
            ref_stage_lok = stgref
         endif
         if ((change_lok_stg .gt. 0.0 .and. (rlkstg .gt.
     $    stgref_lok_to_nstor .and. rlkstg .le. ref_stage_lok))
     $     .or. rlkstg .gt. ref_stage_lok) then
           call northlok_res(1,rnorth_stor_res_area,0.0,10.0
     $      ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
     $      ,actual_et,actual_seepage,res2lok, lok2res, hres, month
     $      ,ifyr, iyear, iday,res_seep_factor)
         elseif (rlkstg .lt. stgref_nstor_to_lok) then
           call northlok_res(-1,rnorth_stor_res_area,0.0,10.0
     $      ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
     $      ,actual_et,actual_seepage,res2lok, lok2res, hres, month
     $      ,ifyr, iyear, iday,res_seep_factor)
         else
           call northlok_res(0,rnorth_stor_res_area,0.0,10.0
     $      ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
     $      ,actual_et,actual_seepage,res2lok, lok2res, hres, month
     $      ,ifyr, iyear, iday,res_seep_factor)
         endif
       else
         pred_change_in_lok_sto = 0.5*climate_index(month)*1.0E6
     $                            - totloketvol3(month)
     $                          - avg_demand_lok3(month)
         month_pred = month + 3
         if (month_pred .gt. 12) month_pred = month_pred - 12
         call bpts(iday,month_pred,10.0,1,n_fc_zones+2,1
     $                      ,rnorth_stor_pred)
         if (lok_split_option .eq. 0) then
           SPRES = STG2STO(rlkstg,0)
           STO_PROJ = SPRES + pred_change_in_lok_sto
           pred_lok_stg = STG2STO(STO_PROJ,1)
         else
           SPRES = STG2STOLITZ(rlkstg,0)
           STO_PROJ = SPRES + pred_change_in_lok_sto
           pred_lok_stg = STG2STOLITZ(STO_PROJ,1)
```

```
    endif
    if ((pred_lok_stg .ge. rnorth_stor_pred
$      .and. itrib_hydro_cond_opt .eq. 1
$      .and. (trib_rf_et .gt.
$      trib_rfet_thres .or. s65e_runff_wkly
$      .ge. s65e_runff_wkly_thres))
$      .or. ((mzone .lt. mzone_puls_top .or. (mzone .ge.
$      mzone_puls_top .and. mzone .le. mzone_puls_bot
$      .and. (stagelo .lt. rnorth_stor_pred
$      .and.climate_index(month) .gt. s_forec_eaar_thres2
$      .or. (stagelo.ge.stgref_lok_to_nstor
$      .and. climate_index(month) .gt. s_forec_eaar_thres1))
$      .and. izone_flex_eaares_opt .eq. 1))
$      .and. iadd_ops_to_eaa_res_line .eq. 1)
$      .or. (nint(S2_state_eaares) .gt. 0 .and.
$      izone_flex_eaares_opt .eq. 0) .or.(pred_lok_stg .gt.
$      rnorth_stor_pred .and. itrib_hydro_cond_opt
$      .eq. 0 .and.izone_flex_eaares_opt.eq.0)) then
       call northlok_res(1,rnorth_stor_res_area,0.0,10.0
$      ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
$      ,actual_et,actual_seepage,res2lok, lok2res, hres, month
$      ,ifyr, iyear, iday,res_seep_factor)
    else
     if (DRYSEAS) then
       call bpts(iday,month_pred,10.0,1,n_fc_zones+5,1
$         ,stgref_nstor_to_lok_pred)
       if (pred_lok_stg .lt. stgref_nstor_to_lok_pred) then
         call northlok_res(-1,rnorth_stor_res_area,0.0,10.0
$         ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
$         ,actual_et,actual_seepage,res2lok, lok2res, hres, month
$         ,ifyr, iyear, iday,res_seep_factor)
       else
         call northlok_res(0,rnorth_stor_res_area,0.0,10.0
$         ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
$         ,actual_et,actual_seepage,res2lok, lok2res, hres, month
$         ,ifyr, iyear, iday,res_seep_factor)
       endif
     else
       if (rlkstg.lt.stgref_nstor_to_lok .and. climate_index(month)
$            .lt. 1.5) then
         call northlok_res(-1,rnorth_stor_res_area,0.0,10.0
$         ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
$         ,actual_et,actual_seepage,res2lok, lok2res, hres, month
$         ,ifyr, iyear, iday,res_seep_factor)
       else
         call northlok_res(0,rnorth_stor_res_area,0.0,10.0
$         ,hres0,norstorrf(jday),norstorpet(jday),4800.,4800.
$         ,actual_et,actual_seepage,res2lok, lok2res, hres, month
$         ,ifyr, iyear, iday,res_seep_factor)
       endif
     endif
    endif
  endif
  hres0 = hres
  RLKSTOR = RLKSTOR + (res2lok - lok2res) * 1.9835
  rnorth_stor_to_lok = res2lok
  rlok_to_north_stor = lok2res
```

```
      endif
      RLKSTG = STG2STO(RLKSTOR,1)
c
c  backpump Caloosahatchee reservoir water into LOK
c  if reservoir stage is above threshold at prescribed
c  rate
c
      cresbp2lok = 0.0
      if (no_puls_zones .gt. 0) then
       if (rlkstg.lt.stgref_pulse_bot .or. (mzone .ge. mzone_puls_top
     $    .and.mzone.le.mzone_puls_bot .and. izone_flex_eaares_opt.eq.1
     $    .and. ((climate_index(month) .lt. rmin_clim_indx_thres
     $    (mzone,2,ioper_opt) .and. iclimate_opt .eq. 1) .and. (
     $    itrib_hydro_cond_opt .eq. 0 .or. (itrib_hydro_cond_opt .eq. 1
     $    .and. (trib_rf_et .le. 8.0.and. s65e_runff_wkly
     $    .le. 9000)))))) then
        if (hres01 .gt. cal_res_dpth_thres_bp) then
           cresbp2lok = cresbp_max_rate*1.9835
           rmax_vol_bp2lok = (hres01 - cal_res_dpth_thres_bp)
     $          * ares01
           cresbp2lok = amin1(cresbp2lok,rmax_vol_bp2lok)
           hres01 = hres01 - cresbp2lok/ares01
           rlkstor = rlkstor + cresbp2lok
        endif
       endif
      else
       if (rlkstg .lt. stgref) then
         if (hres01 .gt. cal_res_dpth_thres_bp) then
           cresbp2lok = cresbp_max_rate*1.9835
           rmax_vol_bp2lok = (hres01 - cal_res_dpth_thres_bp)
     $         * ares01
           cresbp2lok = amin1(cresbp2lok,rmax_vol_bp2lok)
           hres01 = hres01 - cresbp2lok/ares01
           rlkstor = rlkstor + cresbp2lok
         endif
       endif
      endif
      rlkstg = stg2sto(rlkstor,1)
      call bpts(IDAY,MONTH,rlkstg,1,mzone,0,stgref_lkstg)
C
C COMPUTE REGULATORY RELEASES TO ST LUCIE CANAL AND CALOOSAHATCHEE RIVER
C
          REGS308 = 0.0
          REGS77  = 0.0
c
c  determine when LOK inflows from S-65E triggers larger releases
c  to estuaries when only the climate forecasts are used (less than
c  the full WSE decision tree is implemented)
c
          if (itrib_hydro_cond_opt .eq. 0) then
            if (s65ernff(jday) .gt. s65e_inflw_thres_high(iseas)*1.9835
     $       .or. lsim_s65e_clim ) then
              lsim_s65e_clim = .true.
              lsim_s65e_clim = .not. (s65ernff(jday) .lt.
     $          s65e_inflw_thres_low(iseas)*1.9835)

            endif
```

```
            if (lsim_s65e_clim) then
              ctrib_hydro_cond = 'WETCOND'
            else
              ctrib_hydro_cond = 'DRYNORM'
            endif
          else
            lsim_s65e_clim = .false.
          endif
C
C   UPDATE REFERENCE LAKE STAGE ARRAY FOR REGULATORY RELEASES
C
            DO i = 10,2,-1
             rstgpls(i) = rstgpls(i-1)
             do iz = 1,n_fc_zones
              stg_lok_sched (iz,i) = stg_lok_sched (iz,i-1)
             enddo
            enddo
            rstgpls(1) = STAGELO
            do iz = 1,n_fc_zones
             izonez = iz
             call bpts(iday,month,10.0,1,izonez,1,stgref_z)
             stg_lok_sched (iz,1) = stgref_z
            enddo
c
c  compute regulatory releases to tidewater
c
            lo_reg_zone = mzone
c
c  regulatory releases for LOK stages above the pulse zone(s)
c
            IF (mzone .lt. mzone_puls_top .and. ((ipulse .eq. 0
     $        .and. iclimate_opt .eq. 0) .or. iclimate_opt .eq. 1)
     $        .or. (mzone .le. n_fc_zones .and. mzone .gt.
     $           mzone_puls_bot .and. ipulse .eq. 0)) then
             if (iclimate_opt .ne. 1 .or. mzone .eq. 1) then
c
c  if climate forecasting is used,computing for ZONE A only
c
              if (iopt_for_interp(mzone) .eq. 0) then
               KFLO(ifls77) = amax1(rmax_out_capac_wet(mzone,1)
     $            - c43est_met_by_lok/1.9835,0.0) +eqs77/1.9835
               KFLO(ifls308) = amax1(rmax_out_capac_wet(mzone,2)-stlro
     $          - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
              else
               call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
               if (mzone .gt. 1) then
                 call bpts(iday,month,10.0,1,mzone-1,1,stgref_zone_top)
               else
                 stgref_zone_top = stgref_zone + 0.25
               endif
               do i = 1,2
                 if (mzone .eq. mzone_puls_top - 1
     $               .and. no_puls_zones .gt. 0) then
                   capac_bot_zone(i) = avg_pulse_release(mzone_puls_top
     $                             - mzone_puls_bot + 1,i)
                 else
                   capac_bot_zone(i) = rmax_out_capac_wet(mzone+1,i)
```

```
                 endif
                 frac_interp = (RLKSTG - stgref_zone)/((stgref_zone_top
     $             - stgref_zone)*frac_depth_zone(mzone,iseas))
                 frac_interp = amin1(frac_interp,1.0)
                 capac_zone(i) = capac_bot_zone(i)
     $             + frac_interp * (rmax_out_capac_wet(mzone,i)
     $             - capac_bot_zone(i))
               enddo
               KFLO(ifls77) = amax1(capac_zone(1)-c43est_met_by_lok
     $                /1.9835,0.0) + eqs77/1.9835
               KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $            - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
              endif
              REGS308 = KFLO(ifls308)  - eqs308/1.9835
              REGS77  = kflo(ifls77) - eqs77/1.9835
              if (iclimate_opt .eq. 1) clokzone = zoneid_p(mzone)
              rleakance = 0.0
             else
c
c  climate forecasting utilized here for Zones B and C (Zones above pulse
c  release zone)
c
               if (iclimate_opt .eq. 1 ) then
                if (mzone .gt. 1) then
                 clokzone = zoneid_p(mzone)
c
c  consider tributary hydrology as priority which is used to guide the
c  implementation of climate forecasting
c
                 if ((trib_rf_et .gt. trib_rf_et_thres(3
     $              ,mzone) .or. s65e_runff_wkly .gt.
     $              s65e_runff_thres(3,mzone)) .and.itrib_hydro_cond_opt
     $              .eq. 1 .and. (ipulse .eq. 0 .or. (mzone.eq.2 .and.
     $              ipulse.eq.1))) then
c
c  extremely wet conditions for LOK tributary inflow region
c
                    ctrib_hydro_cond = cgen_trib_hydro_categ(3,mzone)
                     if (climate_index(month).ge.rmin_clim_indx_thres
     $                 (mzone,2,ioper_opt)) then
                       cseas_forec = cgen_seas_categ(3,mzone,ioper_opt)
                       kflo(ifls77) =amax1(rmax_out_capac_wet(mzone-1,1)
     $                   - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                       kflo(ifls308) = amax1(rmax_out_capac_wet(mzone-1
     $                   ,2) - stlro - sledmd_met_by_lok/1.9835,0.0)
     $                   + eqs308/1.9835
                     else
                       cseas_forec = cgen_seas_categ(2,mzone,ioper_opt)
                       if (iopt_for_interp(mzone) .eq. 0) then
                         kflo(ifls77) =amax1(rmax_out_capac_wet(mzone,1)
     $                     - c43est_met_by_lok/1.9835,0.0) +eqs77/1.9835
                         kflo(ifls308) = amax1(rmax_out_capac_wet(mzone
     $                    ,2) - stlro - sledmd_met_by_lok/1.9835,0.0)
     $                    + eqs308/1.9835
                       else
                        call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
                        call bpts(iday,month,10.0,1,mzone-1,1
```

```
     $                     ,stgref_zone_top)
                        do i = 1,2
                         if (mzone .eq. mzone_puls_top - 1
     $                    .and. no_puls_zones .gt. 0) then
                          capac_bot_zone(i) = avg_pulse_release
     $                         (ipulse_level_in_zone(mzone),i)
                         else
                          capac_bot_zone(i)=rmax_out_capac_wet(mzone+1
     $                         ,i)
                         endif
                         frac_interp = (RLKSTG - stgref_zone)
     $                    /((stgref_zone_top- stgref_zone)
     $                 *frac_depth_zone(mzone,iseas))
                         frac_interp = amin1(frac_interp,1.0)
                         capac_zone(i) = capac_bot_zone(i)
     $                    + frac_interp * (rmax_out_capac_wet(mzone,i)
     $                    - capac_bot_zone(i))
                        enddo
                        KFLO(ifls77) = amax1(capac_zone(1)
     $                    - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                        KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $                    - sledmd_met_by_lok/1.9835,0.0) +eqs308/1.9835
                        rleakance = 0.0
                      endif
                    endif
                    if (mzone.eq.2.and.ipulse.eq.1) ipulse = 0
                    REGS308 = KFLO(ifls308)  - eqs308/1.9835
                    REGS77  = kflo(ifls77) - eqs77/1.9835
                  elseif ((trib_rf_et .gt. trib_rf_et_thres
     $             (2,mzone) .or. s65e_runff_wkly .gt.
     $             s65e_runff_thres(2,mzone)) .and. itrib_hydro_cond_opt
     $             .eq. 1 .and. (ipulse .eq. 0 .or. (mzone.eq.2 .and.
     $             ipulse.eq.1))) then
c
c  normal to very wet(ZONE B) or wet to very wet(ZONE C) tributary
c  hydrologic conditions
c
                    ctrib_hydro_cond = cgen_trib_hydro_categ(2,mzone)
                    if (ioper_opt .eq. 2 .and. iopt_drawdown .eq. 1
     $              .and. multi_seas_pred_opt .eq. 1 .and. ipulse.eq.0
     $              .and. rlkstg .lt. zonec_norm) then
                      if (rmulti_seas_pred(month) .lt.
     $                rmulti_seas_thres(2,mzone,ioper_opt)) then
                         kflo(ifls77)  = eqs77/1.9835 + 5.0
                         kflo(ifls308) = eqs308/1.9835
                         rleakance = 5.0
                         cmulti_seas_forec = cgen_multi_seas_categ(1
     $                   ,mzone,ioper_opt)
                         go to 1000
                      endif
                    endif
                    if (iopt_for_interp(mzone) .eq. 0) then
                      kflo(ifls77) = amax1(rmax_out_capac_wet(mzone,1)
     $                  - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                      kflo(ifls308) = amax1(rmax_out_capac_wet(mzone,2)
     $                  - sledmd_met_by_lok/1.9835 - stlro,0.0) + eqs308
     $                  / 1.9835
```

```
                    else
                      call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
                      call bpts(iday,month,10.0,1,mzone-1,1
     $                      ,stgref_zone_top)
                      do i = 1,2
                       if (mzone .eq. mzone_puls_top - 1
     $                   .and. no_puls_zones .gt. 0) then
                         capac_bot_zone(i) = avg_pulse_release
     $                          (ipulse_level_in_zone(mzone),i)
                       else
                         capac_bot_zone(i) = rmax_out_capac_wet(mzone+1
     $                          ,i)
                       endif
                       frac_interp = (RLKSTG - stgref_zone)
     $                   /((stgref_zone_top- stgref_zone)
     $               *frac_depth_zone(mzone,iseas))
                       frac_interp = amin1(frac_interp,1.0)
                       capac_zone(i) = capac_bot_zone(i)
     $                  + frac_interp * (rmax_out_capac_wet(mzone,i)
     $                  - capac_bot_zone(i))
                      enddo
                      KFLO(ifls77)=amax1(capac_zone(1)-c43est_met_by_lok
     $                   /1.9835,0.0) + eqs77/1.9835
                      KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $                   - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                      rleakance = 0.0
                    endif
                    if (mzone.eq.2.and.ipulse.eq.1) ipulse = 0
                    REGS308 = KFLO(ifls308)  - eqs308/1.9835
                    REGS77  = kflo(ifls77) - eqs77/1.9835
                  else
c
c  dry (ZONE B) or normal to dry (ZONE C) tributary hydrologic conditions
c
                   if (itrib_hydro_cond_opt .eq. 1) then
                     ctrib_hydro_cond = cgen_trib_hydro_categ(1,mzone)
                   endif
c
                   if (ioper_opt .eq. 2 .and. iopt_drawdown .eq. 1
     $              .and. multi_seas_pred_opt .eq. 1 .and.
     $              rlkstg .lt. zonec_norm .and.ipulse .eq. 0) then
                     if (rmulti_seas_pred(month) .lt.
     $               rmulti_seas_thres(2,mzone,ioper_opt)) then
                        kflo(ifls77)  = eqs77/1.9835 + 5.0
                        kflo(ifls308) = eqs308/1.9835
                        rleakance = 5.0
                        cmulti_seas_forec =cgen_multi_seas_categ(1,mzone
     $                   ,ioper_opt)
                        go to 1000
                     endif
                   endif
c
c    LOK stage conditions in ZONE B
c
                   if (mzone .eq. 2 .and. (stagelo .ge. 17.5 .or.
     $               .not. DRYSEAS)) then
c
```

```
c      release steady flow here if LOK stage is in Zone B and we are in
c      the wet season or LOK stage is above 17.5 ft during the dry season.
c
                    cstg_lok_pres = 'GE_17.5_OR_WETS'
                    if (iopt_for_interp(mzone) .eq. 0) then
                      kflo(ifls77) = amax1(rmax_out_capac_wet(mzone,1)
     $                 - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                      kflo(ifls308) = amax1(rmax_out_capac_wet(mzone,2)
     $                 - sledmd_met_by_lok/1.9835 - stlro,0.0) + eqs308
     $                 / 1.9835
                    else
                      call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
                      call bpts(iday,month,10.0,1,mzone-1,1
     $                      ,stgref_zone_top)
                      do i = 1,2
                       if (mzone .eq. mzone_puls_top - 1
     $                   .and. no_puls_zones .gt. 0) then
                         capac_bot_zone(i) = avg_pulse_release
     $                          (ipulse_level_in_zone(mzone),i)
                       else
                         capac_bot_zone(i) = rmax_out_capac_wet(mzone+1
     $                          ,i)
                       endif
                       frac_interp = (RLKSTG - stgref_zone)
     $                   /((stgref_zone_top- stgref_zone)
     $              *frac_depth_zone(mzone,iseas))
                       frac_interp = amin1(frac_interp,1.0)
                       capac_zone(i) = capac_bot_zone(i)
     $                  + frac_interp * (rmax_out_capac_wet(mzone,i)
     $                  - capac_bot_zone(i))
                      enddo
                      KFLO(ifls77) = amax1(capac_zone(1) -
     $                  c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                      KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $                  - sledmd_met_by_lok/1.9835,0.0)+ eqs308/1.9835
                    endif
                    rleakance = 0.0
                    REGS308 = KFLO(ifls308)  - eqs308/1.9835
                    REGS77  = kflo(ifls77) - eqs77/1.9835
                    if (mzone.eq.2.and.ipulse.eq.1) ipulse = 0
                  else
c
c
                  if (mzone .eq. 2) cstg_lok_pres = 'LT_17.5_AND_DRYS'
                  if ((( HURRPRED .and. no_of_hurr_pred .gt.
     $            no_pred_storms_thres) .or. (lsim_s65e_clim
     $            .and. mzone .eq. 2) .or.
     $            climate_index(month).ge.rmin_clim_indx_thres(mzone,
     $            2,ioper_opt)) .and. (ipulse .eq. 0.or.(mzone.eq.2.and.
     $            ipulse.eq.1))) then
c
c     seasonal outlook is wet to very wet : steady flow if LOK stage is
c     in ZONE B or if not in 10-day pulse cycle,in ZONE C or in additional
c     zones ABOVE the pulse release zone(s)
c
                    if (climate_index(month).ge.rmin_clim_indx_thres
     $               (mzone,2,ioper_opt)) then
```

```
                      cseas_forec = cgen_seas_categ(3,mzone,ioper_opt)
                    endif
                    if (iopt_for_interp(mzone) .eq. 0) then
                      kflo(ifls77) = amax1(rmax_out_capac_wet(mzone,1)
     $                  - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                      kflo(ifls308) = amax1(rmax_out_capac_wet(mzone,2)
     $                  - sledmd_met_by_lok/1.9835 - stlro,0.0) + eqs308
     $                  / 1.9835
                    else
                      call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
                      call bpts(iday,month,10.0,1,mzone-1,1
     $                      ,stgref_zone_top)
                      do i = 1,2
                       if (mzone .eq. mzone_puls_top - 1
     $                   .and. no_puls_zones .gt. 0) then
                        capac_bot_zone(i) = avg_pulse_release
     $                         (ipulse_level_in_zone(mzone),i)
                       else
                        capac_bot_zone(i) = rmax_out_capac_wet(mzone+1
     $                         ,i)
                       endif
                       frac_interp = (RLKSTG - stgref_zone)
     $                   /((stgref_zone_top- stgref_zone)
     $               *frac_depth_zone(mzone,iseas))
                       frac_interp = amin1(frac_interp,1.0)
                       capac_zone(i) = capac_bot_zone(i)
     $                  + frac_interp * (rmax_out_capac_wet(mzone,i)
     $                  - capac_bot_zone(i))
                      enddo
                      KFLO(ifls77) = amax1(capac_zone(1)
     $                  - c43est_met_by_lok/1.9835,0.0)  + eqs77/1.9835
                      KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $                  - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                    endif
                    rleakance = 0.0
                    if (mzone.eq.2.and.ipulse.eq.1) ipulse = 0
c
                  elseif ((climate_index(month) .ge.
     $                rmin_clim_indx_thres(mzone,1,ioper_opt) .and.
     $                mzone .gt. 2 .and. multi_seas_pred_opt .eq. 1)
     $                .and. rmulti_seas_pred(month) .gt.
     $                rmulti_seas_thres(3,mzone,ioper_opt)) then
c
c  Code for ZONE C and other ZONES lower than ZONE C that are above
c  pulse zones. The multi-seasonal outlook is sufficiently high for
c  steady releases during times of LOK drawdown.
c
                    cseas_forec = cgen_seas_categ(2,mzone,ioper_opt)
                    cmulti_seas_forec = cgen_multi_seas_categ(3,mzone
     $               ,ioper_opt)

                    if (iopt_for_interp(mzone) .eq. 0) then
                      kflo(ifls77) = amax1(rmax_out_capac_wet(mzone,1)
     $                  - c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                      kflo(ifls308) = amax1(rmax_out_capac_wet(mzone,2)
     $                  - sledmd_met_by_lok/1.9835 - stlro,0.0) + eqs308
     $                  / 1.9835
```

```
                    else
                      call bpts(iday,month,10.0,1,mzone,1,stgref_zone)
                      call bpts(iday,month,10.0,1,mzone-1,1
     $                      ,stgref_zone_top)
                      do i = 1,2
                       if (mzone .eq. mzone_puls_top - 1
     $                   .and. no_puls_zones .gt. 0) then
                         capac_bot_zone(i) = avg_pulse_release
     $                          (ipulse_level_in_zone(mzone),i)
                       else
                         capac_bot_zone(i) = rmax_out_capac_wet(mzone+1
     $                          ,i)
                       endif
                       frac_interp = (RLKSTG - stgref_zone)
     $                   /((stgref_zone_top- stgref_zone)
     $               *frac_depth_zone(mzone,iseas))
                       frac_interp = amin1(frac_interp,1.0)
                       capac_zone(i) = capac_bot_zone(i)
     $                  + frac_interp * (rmax_out_capac_wet(mzone,i)
     $                  - capac_bot_zone(i))
                      enddo
                      KFLO(ifls77) = amax1(capac_zone(1)
     $                  - c43est_met_by_lok/1.9835,0.0)  + eqs77/1.9835
                      KFLO(ifls308) = amax1(capac_zone(2)-stlro
     $                  - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                    endif
c
                  elseif ((climate_index(month) .ge.
     $                rmin_clim_indx_thres(mzone,1,ioper_opt) .and.
     $                multi_seas_pred_opt .eq. 0) .or.(((mzone
     $                .gt. 2 .and. climate_index(month) .ge.
     $                rmin_clim_indx_thres(mzone,1,ioper_opt) .and.
     $                rmulti_seas_pred(month).ge.rmulti_seas_thres(1
     $                ,mzone,ioper_opt)).or.mzone.eq.2) .and.
     $                multi_seas_pred_opt .eq.1) .or.ipulse .eq. 1) then
c
c  seasonal outlook calls for pulse releases in ZONE B and if multi-seasonal
c  outlook is sufficiently high,pulse releases in ZONE C. Option exists to
c  release steady flow which could be different from that released for
c  wet forecast
c
                    IF (ipulse .eq. 0) THEN
                      cseas_forec = cgen_seas_categ(2,mzone,ioper_opt)
                      if(multi_seas_pred_opt .eq. 1 .and. mzone .gt. 2)
     $                  then
                        if (rmulti_seas_pred(month).ge.rmulti_seas_thres
     $                  (2,mzone,ioper_opt)) then
                           cmulti_seas_forec = cgen_multi_seas_categ
     $                     (2,mzone,ioper_opt)
                           iplslevel = ipulse_level_in_zone(mzone)
                        else
                           cmulti_seas_forec = cgen_multi_seas_categ
     $                     (1,mzone,ioper_opt)
                           iplslevel = ipulse_level_multi_seas(1,mzone
     $                     ,ioper_opt)
                        endif
                      else
```

```
                        iplslevel = ipulse_level_in_zone(mzone)
                      endif
                    ELSE
c                   if (climate_index(month) .lt.
c    $                rmin_clim_indx_thres(mzone,1,ioper_opt) .or.
c    $                (rmulti_seas_pred(month) .lt.
c    $                rmulti_seas_thres(1,mzone,ioper_opt).and.
c    $                multi_seas_pred_opt.eq.1) .or.climate_index(month)
c    $                .ge. rmin_clim_indx_thres(mzone,2,ioper_opt)) then
                       cseas_forec = 'PULSE C'
                       if (multi_seas_pred_opt .eq. 1)
     $                  cmulti_seas_forec = 'PULSE C'
                    ENDIF
                    if (opt_for_pulsing(mzone) .eq. 'PULSE  '
     $                  .or. ipulse .eq. 1) then
c
                      if (izone_flex_eaares_opt.eq.0 .or.
     $                 (izone_flex_eaares_opt.eq.1 .and.
     $                 ((ieaarsn_no.le.ntotres .and.
     $                 stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
     $                 (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
     $                 .and.ieaarss_no .le.ntotres)) .or.
     $                 ipulse.eq.1)) then
                        call qpulse(rstgpls,RLKSTG,iplslevel
     $                   ,slpulse,calpulse,month,iday,mzone)
                        rleakance = 0.0
                      else
                        calpulse = 0.0
                        slpulse  = 0.0
                        rleakance = 5.0
                      endif
                      KFLO(ifls77) = amax1(calpulse - cal_rnff_frm_reg
     $                 - c43est_met_by_lok/1.9835,0.0)+rleakance+eqs77
     $                 / 1.9835
                      if (stagelo .ge. stg_backflw_stl) then
                        KFLO(ifls308)= amax1(slpulse-stlro
     $                   - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                      else
                        KFLO(ifls308)= amax1(slpulse - sledmd_met_by_lok
     $                      /1.9835,0.0) + eqs308/1.9835
                      endif
                    else
                      if (izone_flex_eaares_opt.eq.0 .or.
     $                 (izone_flex_eaares_opt.eq.1 .and.
     $                 ((ieaarsn_no.le.ntotres .and.
     $                 stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
     $                 (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
     $                 .and.ieaarss_no .le.ntotres)) .or.
     $                 ipulse.eq.1)) then

                        KFLO(ifls77) =amax1(rmax_out_capac_norm(mzone,1)
     $                    -c43est_met_by_lok/1.9835,0.0) + eqs77/1.9835
                        KFLO(ifls308) = amax1(rmax_out_capac_norm(mzone
     $                       ,2) - stlro - sledmd_met_by_lok/1.9835,0.0)
     $                       + eqs308/1.9835
                        rleakance = 0.0
                      else
```

```
                      rleakance = 5.0
                      slpulse  = 0.0
                      calpulse = 0.0
                      KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
     $                  /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
     $                  + rleakance
                      KFLO(ifls308) = amax1(slpulse - stlro
     $                  - sledmd_met_by_lok/1.9835,0.0) +eqs308/1.9835
                    endif
                  endif
                else
                  if (climate_index(month) .lt.
     $              rmin_clim_indx_thres(mzone,1,ioper_opt)) then
                    cseas_forec = cgen_seas_categ(1,mzone,ioper_opt)
                  else
                    cseas_forec = cgen_seas_categ(2,mzone,ioper_opt)
                  endif
                  if (multi_seas_pred_opt .eq. 1) then
                   if (rmulti_seas_pred(month) .lt.
     $              rmulti_seas_thres(1,mzone,ioper_opt)) then
                    cmulti_seas_forec = 'EXT DRY'
                   else
                    cmulti_seas_forec = cgen_multi_seas_categ(1,mzone
     $              ,ioper_opt)
                   endif
                  endif
                  calpulse = 0.0
                  slpulse  = 0.0
                  rleakance = 5.0
                  KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
     $              /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
     $              + rleakance
                  KFLO(ifls308) = amax1(slpulse - stlro
     $              - sledmd_met_by_lok/1.9835,0.0) +eqs308/1.9835
                endif
                REGS308 = KFLO(ifls308)  - eqs308/1.9835
                REGS77  = kflo(ifls77) - eqs77/1.9835 - rleakance
               endif
               endif
              endif
              endif
             endif
           ELSEIF (((mzone .ge. mzone_puls_top .AND. mzone .LE.
     $       mzone_puls_bot) .or. ipulse .eq. 1) .and.
     $       no_puls_zones .gt. 0) then
c
c LOK stages in pulse release zone(s) - e.g. ZONE D in WSE Schedule
c
              if (iclimate_opt .eq. 0) then
                if (ipulse .eq. 0) iplslevel =
     $                     ipulse_level_in_zone(mzone)
              else
                if (ipulse .eq. 0) then
                 if (itrib_hydro_cond_opt.eq.1) then
                  if (s65e_runff_wkly.gt.
     $             s65e_runff_thres(3,mzone) .or. trib_rf_et
     $             .gt. trib_rf_et_thres(3,mzone)) then
```

```
                    iplslevel = ipulse_level_trib_hyd(4,mzone
     $              ,ioper_opt)
                   elseif(s65e_runff_wkly.gt.
     $              s65e_runff_thres(2,mzone) .or. trib_rf_et
     $              .gt. trib_rf_et_thres(2,mzone)) then
                    iplslevel = ipulse_level_trib_hyd(3,mzone
     $              ,ioper_opt)
                   elseif(s65e_runff_wkly.gt.
     $              s65e_runff_thres(1,mzone) .or. trib_rf_et
     $              .gt. trib_rf_et_thres(1,mzone)) then
                    iplslevel = ipulse_level_trib_hyd(2,mzone
     $              ,ioper_opt)
                   else
                    iplslevel = ipulse_level_trib_hyd(1,mzone
     $              ,ioper_opt)
                   endif
                  else
                    if (multi_seas_pred_opt.eq.1) then
                      if (rmulti_seas_pred(month).gt.
     $                rmulti_seas_thres(3,mzone,ioper_opt)) then
                        iplslevel = ipulse_level_multi_seas(3
     $                   ,mzone,ioper_opt)
                      else
                        iplslevel = ipulse_level_multi_seas(2
     $                   ,mzone,ioper_opt)
                      endif
                    else
                      iplslevel = ipulse_level_in_zone(mzone)
                    endif
                  endif
                 endif
                 if (mzone .ge. mzone_puls_top .AND. mzone .LE.
     $            mzone_puls_bot) then
                   clokzone = zoneid_p(mzone)
                 else
                   clokzone = 'BEL_SCH'
                 endif
               endif
               if (iclimate_opt.eq.1.and.(itrib_hydro_cond_opt.eq.1
     $            .and.((s65e_runff_wkly .gt.
     $            s65e_runff_thres(3,mzone) .or. trib_rf_et
     $            .gt. trib_rf_et_thres(3,mzone)) .or.
     $            ipulse .eq. 1))) then
c
c    Tributary hydrologic conditions extremely wet
c
                  if (mzone .ge. mzone_puls_top .and. mzone .le.
     $               mzone_puls_bot) then
                   if (s65e_runff_wkly .gt.
     $              s65e_runff_thres(3,mzone) .or. trib_rf_et
     $              .gt. trib_rf_et_thres(3,mzone)) then
                    ctrib_hydro_cond = cgen_trib_hydro_categ(4,mzone)
                   elseif(s65e_runff_wkly .gt.
     $              s65e_runff_thres(2,mzone) .or. trib_rf_et
     $              .gt. trib_rf_et_thres(2,mzone)) then
                    ctrib_hydro_cond = cgen_trib_hydro_categ(3,mzone)
                   elseif(s65e_runff_wkly.gt.
```

```
     $                s65e_runff_thres(1,mzone) .or. trib_rf_et
     $                .gt. trib_rf_et_thres(1,mzone)) then
                      ctrib_hydro_cond = cgen_trib_hydro_categ(2,mzone)
                     else
                      ctrib_hydro_cond = cgen_trib_hydro_categ(1,mzone)
                     endif
                    endif
                    call bpts(iday,month,10.0,1,mzone_puls_top-1,1
     $                      ,stgref_zonec)
c
c    WSE LOK stage criteria
c
                    if (rlkstg .gt. stgref_zonec - 0.5) then
c
c      LOK stage within 0.5 ft of ZONE C
c
                       if (ipulse .eq. 0) then
                         cstg_lok_pres = 'WITHIN_0.5_FT_ZONEC'
                       endif
                       if (climate_index(month) .gt. rmin_clim_indx_thres
     $                    (mzone,2,ioper_opt) .and. ipulse .eq. 0) then
                         cseas_forec = cgen_seas_categ(3,mzone,ioper_opt)
                         kflo(ifls77) = amax1(rmax_out_capac_wet(mzone-1
     $                     ,1) - c43est_met_by_lok/1.9835,0.0) + eqs77
     $                     /1.9835
                         kflo(ifls308) = amax1(rmax_out_capac_wet(mzone-1
     $                     ,2) - sledmd_met_by_lok/1.9835 - stlro,0.0)
     $                     + eqs308/1.9835
                         REGS308 = KFLO(ifls308)  - eqs308/1.9835
                         REGS77  = kflo(ifls77) - eqs77/1.9835
                       else
                         if (ipulse .eq. 0) then
                           if (climate_index(month) .le.
     $                     rmin_clim_indx_thres(mzone,2,ioper_opt)) then
                            cseas_forec = cgen_seas_categ(2,mzone
     $                      ,ioper_opt)
                           endif
                         else
                           cseas_forec = 'PULSE_C'
                           cmulti_seas_forec = 'PULSE_C'
                         endif
                         if (izone_flex_eaares_opt.eq.0 .or.
     $                    (izone_flex_eaares_opt.eq.1 .and.
     $                    ((ieaarsn_no.le.ntotres .and.
     $                    stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
     $                    (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
     $                    .and.ieaarss_no .le.ntotres)) .or.
     $                    ipulse.eq.1)) then
                           call qpulse(rstgpls,RLKSTG,iplslevel
     $                      ,slpulse,calpulse,month,iday,mzone)
                           rleakance = 0.0
                         else
                           calpulse = 0.0
                           slpulse  = 0.0
                           rleakance = 5.0
                         endif
                         KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
```

```
     $                   /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
                       if (stagelo .ge. stg_backflw_stl) then
                         KFLO(ifls308)= amax1(slpulse-stlro
     $                    - sledmd_met_by_lok/1.9835,0.0)+eqs308/1.9835
                         REGS308 = amax1(slpulse - stlro
     $                    - sledmd_met_by_lok/1.9835,0.0)
                       else
                         KFLO(ifls308)= amax1(slpulse -
     $                     sledmd_met_by_lok/1.9835,0.0) +eqs308/1.9835
                         REGS308 = amax1(slpulse-sledmd_met_by_lok
     $                                 /1.9835,0.0)
                       endif
                       REGS77  = amax1(calpulse-c43est_met_by_lok
     $                   /1.9835-cal_rnff_frm_reg,0.0)
                     endif
                   else
c
c      LOK stage NOT within 0.5 ft of ZONE C
c
                     if (ipulse .eq. 0) then
                       if (mzone .ge. mzone_puls_top .and. mzone .le.
     $                  mzone_puls_bot) then
                        cstg_lok_pres = 'NOT_WITH_.5_FT_ZONEC'
                       endif
                     endif
                     if((rmulti_seas_pred(month).gt.rmulti_seas_thres(2
     $                ,mzone,ioper_opt).and.multi_seas_pred_opt.eq.1)
     $                .or. ipulse .eq. 1) then
                       if (multi_seas_pred_opt.eq.1) then
                        if (mzone .le. mzone_puls_bot
     $                   .and. ipulse .eq. 0) then
                         if (rmulti_seas_pred(month).gt.
     $                    rmulti_seas_thres(2,mzone,ioper_opt).and.
     $                    multi_seas_pred_opt.eq.1) then
                          cmulti_seas_forec = cgen_multi_seas_categ(2
     $                        ,mzone,ioper_opt)
                          if (rmulti_seas_pred(month).gt.
     $                      rmulti_seas_thres(3,mzone,ioper_opt)) then
                            iplslevel = ipulse_level_multi_seas(3
     $                       ,mzone,ioper_opt)
                          else
                            iplslevel = ipulse_level_multi_seas(2
     $                       ,mzone,ioper_opt)
                          endif
                         endif
                        else
                          cmulti_seas_forec = 'PULSE_C'
                          cseas_forec = 'PULSE_C'
                        endif
                       endif
                       if (izone_flex_eaares_opt.eq.0 .or.
     $                  (izone_flex_eaares_opt.eq.1 .and.
     $                  ((ieaarsn_no.le.ntotres .and.
     $                  stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
     $                  (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
     $                  .and.ieaarss_no .le.ntotres)) .or.
     $                  ipulse.eq.1)) then
```

```
                      call qpulse(rstgpls,RLKSTG,iplslevel
$                     ,slpulse,calpulse,month,iday,mzone)
                      rleakance = 0.0
                    else
                      calpulse = 0.0
                      slpulse  = 0.0
                      rleakance = 5.0
                    endif
                  elseif ((rmulti_seas_pred(month).gt.
$                 rmulti_seas_thres(1,mzone,ioper_opt)
$                 .and.multi_seas_pred_opt.eq.1) .or.
$                 ipulse .eq. 1) then
                  if (multi_seas_pred_opt.eq.1) then
                     if (mzone .le. mzone_puls_bot
$                     .and. ipulse .eq. 0) then
                      if (rmulti_seas_pred(month).gt.
$                      rmulti_seas_thres(1,mzone,ioper_opt).and.
$                      multi_seas_pred_opt.eq.1) then
                      cmulti_seas_forec = cgen_multi_seas_categ(1
$                         ,mzone,ioper_opt)
                      iplslevel = ipulse_level_multi_seas(1,mzone,
$                      ioper_opt)
                      endif
                     else
                      cmulti_seas_forec = 'PULSE C'
                      cseas_forec = 'PULSE C'
                     endif
                    call qpulse(rstgpls,RLKSTG,iplslevel
$                     ,slpulse,calpulse,month,iday,mzone)
                    rleakance = 0.0
                    if (nint(calpulse).eq.0) rleakance = 5.0
                  endif
                else
                  calpulse = 0.0
                  slpulse  = 0.0
                  rleakance = 5.0
                endif
                KFLO(ifls77) = amax1(calpulse -c43est_met_by_lok
$                    /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
$                    + rleakance
                if (stagelo .ge. stg_backflw_stl) then
                   KFLO(ifls308)= amax1(slpulse-stlro
$                    - sledmd_met_by_lok/1.9835,0.0)+eqs308/1.9835
                   REGS308 = amax1(slpulse - stlro
$                    - sledmd_met_by_lok/1.9835 ,0.0)
                 else
                   KFLO(ifls308)= amax1(slpulse-sledmd_met_by_lok
$                    /1.9835,0.0) + eqs308/1.9835
                   REGS308 = amax1(slpulse-sledmd_met_by_lok
$                    /1.9835,0.0)
                 endif
                REGS77  = amax1(calpulse - c43est_met_by_lok
$                    /1.9835 - cal_rnff_frm_reg,0.0)
              endif
            elseif (iclimate_opt.eq.1.and.(itrib_hydro_cond_opt.eq.1
$             .and.((s65e_runff_wkly .gt.
$             s65e_runff_thres(2,mzone) .or. trib_rf_et
```

```
     $              .gt. trib_rf_et_thres(2,mzone)) .or.
     $              ipulse .eq. 1))) then
c
c     Tributary Hydrologic conditions wet to very wet
c
                   if (mzone .ge. mzone_puls_top .and. mzone .le.
     $                mzone_puls_bot) then
                    ctrib_hydro_cond = cgen_trib_hydro_categ(3,mzone)
                   endif
                   if ((rmulti_seas_pred(month) .gt. rmulti_seas_thres(2
     $                ,mzone,ioper_opt).or.ipulse.eq.1).and.
     $                multi_seas_pred_opt.eq.1) then
                     if (mzone .ge. mzone_puls_top .and.
     $                mzone.le.mzone_puls_bot .and. ipulse .eq. 0) then
                      if (rmulti_seas_pred(month).gt.
     $                  rmulti_seas_thres(3,mzone,ioper_opt)) then
                        iplslevel = ipulse_level_multi_seas(3
     $                   ,mzone,ioper_opt)
                      else
                        iplslevel = ipulse_level_multi_seas(2
     $                   ,mzone,ioper_opt)
                      endif
                      cmulti_seas_forec = cgen_multi_seas_categ(2,mzone
     $                ,ioper_opt)
                     else
                      cmulti_seas_forec = 'PULSE_C'
                     endif
                     if (izone_flex_eaares_opt.eq.0 .or.
     $                (izone_flex_eaares_opt.eq.1 .and.
     $                ((ieaarsn_no.le.ntotres .and.
     $                stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
     $                (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
     $                .and.ieaarss_no .le.ntotres)) .or.
     $                ipulse.eq.1)) then
                      call qpulse(rstgpls,RLKSTG,iplslevel
     $                  ,slpulse,calpulse,month,iday,mzone)
                      rleakance = 0.0
                     else
                      calpulse = 0.0
                      slpulse  = 0.0
                      rleakance = 5.0
                     endif
                   elseif ((rmulti_seas_pred(month).gt.
     $               rmulti_seas_thres(1,mzone,ioper_opt)
     $               .and.multi_seas_pred_opt.eq.1) .or.
     $               ipulse .eq. 1) then
                     if (multi_seas_pred_opt.eq.1) then
                        if (mzone .le. mzone_puls_bot
     $                  .and. ipulse .eq. 0) then
                       if (rmulti_seas_pred(month).gt.
     $                  rmulti_seas_thres(1,mzone,ioper_opt).and.
     $                  multi_seas_pred_opt.eq.1) then
                       cmulti_seas_forec = cgen_multi_seas_categ(1
     $                     ,mzone,ioper_opt)
                       iplslevel = ipulse_level_multi_seas(1,mzone,
     $                  ioper_opt)
                       endif
```

```
                      else
                        cmulti_seas_forec = 'PULSE C'
                        cseas_forec = 'PULSE C'
                      endif
                     call qpulse(rstgpls,RLKSTG,iplslevel
     $                ,slpulse,calpulse,month,iday,mzone)
                     rleakance = 0.0
                     if (nint(calpulse).eq.0) rleakance = 5.0
                   endif
                 else
                   calpulse = 0.0
                   slpulse  = 0.0
                   rleakance = 5.0
                 endif
                 KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
     $              /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
     $              + rleakance
                 if (stagelo .ge. stg_backflw_stl) then
                    KFLO(ifls308)= amax1(slpulse-stlro
     $               -sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                    REGS308 = amax1(slpulse - stlro
     $               -sledmd_met_by_lok/1.9835 ,0.0)
                 else
                    KFLO(ifls308)= amax1(slpulse-sledmd_met_by_lok
     $               /1.9835,0.0) + eqs308/1.9835
                    REGS308 = amax1(slpulse - sledmd_met_by_lok
     $               /1.9835,0.0)
                 endif
                 REGS77  = amax1(calpulse-c43est_met_by_lok
     $               /1.9835-cal_rnff_frm_reg,0.0)
               elseif (iclimate_opt.eq.0.or.(itrib_hydro_cond_opt.eq.1
     $           .and. (s65e_runff_wkly .gt.
     $           s65e_runff_thres(1,mzone) .or. trib_rf_et
     $            .gt. trib_rf_et_thres(1,mzone))) .or.
     $           itrib_hydro_cond_opt.eq.0 .or. ipulse.eq.1) then
c
c     Tributary Conditions are Normal
c
                    if(itrib_hydro_cond_opt.eq.1) then
                       if (mzone .ge. mzone_puls_top .and.
     $                    mzone .le. mzone_puls_bot) then
                        ctrib_hydro_cond=cgen_trib_hydro_categ(2,mzone)
                       endif
                    endif
c
c       Check LOK stage criteria
c
                    call bpts(iday,month,10.0,1,mzone_puls_top-1,1
     $                     ,stgref_zonec)
                    if (rlkstg .gt. stgref_zonec - 0.5
     $                 .and. iclimate_opt .eq. 1 .and.
     $                 multi_seas_pred_opt.eq. 1) then
c
c         LOK stage within 0.5 ft Zone C
c
                      if (mzone .ge. mzone_puls_top .and.
     $                 mzone .le. mzone_puls_bot) then
```

```
                      cstg_lok_pres = 'WITHIN_0.5_FT_ZONEC'
                    endif
                    if ((rmulti_seas_pred(month) .gt.
$                     rmulti_seas_thres(2,mzone,ioper_opt) .and.
$                     multi_seas_pred_opt.eq. 1) .or.ipulse.eq.1)
$                      then
                       if (multi_seas_pred_opt.eq. 1) then
                         if (ipulse .eq. 0) then
                            if (rmulti_seas_pred(month).gt.
$                             rmulti_seas_thres(3,mzone,ioper_opt))
$                             then
                                iplslevel = ipulse_level_multi_seas(3
$                                ,mzone,ioper_opt)
                            else
                                iplslevel = ipulse_level_multi_seas(2
$                                ,mzone,ioper_opt)
                            endif
                         endif
                         if (mzone .ge. mzone_puls_top .and.
$                          mzone .le. mzone_puls_bot) then
                          if (rmulti_seas_pred(month) .gt.
$                         rmulti_seas_thres(2,mzone,ioper_opt)) then
                          cmulti_seas_forec = cgen_multi_seas_categ
$                          (2,mzone,ioper_opt)
                           if (rmulti_seas_pred(month).gt.
$                           rmulti_seas_thres(3,mzone,ioper_opt))
$                           then
                                iplslevel = ipulse_level_multi_seas(3
$                                ,mzone,ioper_opt)
                           else
                                iplslevel = ipulse_level_multi_seas(2
$                                ,mzone,ioper_opt)
                           endif
                          else
                           cmulti_seas_forec = 'PULSE_C'
                          endif
                         else
                          cmulti_seas_forec = 'PULSE_C'
                         endif
                       endif
                       if (izone_flex_eaares_opt.eq.0 .or.
$                      (izone_flex_eaares_opt.eq.1 .and.
$                      ((ieaarsn_no.le.ntotres .and.
$                      stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
$                      (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
$                      .and.ieaarss_no .le.ntotres)) .or.
$                      ipulse.eq.1)) then
                         call qpulse(rstgpls,RLKSTG,iplslevel
$                         ,slpulse,calpulse,month,iday,mzone)
                         rleakance = 0.0
                       else
                         calpulse = 0.0
                         slpulse  = 0.0
                         rleakance = 5.0
                       endif
                    elseif ((rmulti_seas_pred(month).gt.
$                    rmulti_seas_thres(1,mzone,ioper_opt)
```

```
     $                   .and.multi_seas_pred_opt.eq.1) .or.
     $                   ipulse .eq. 1) then
                         if (multi_seas_pred_opt.eq.1) then
                            if (mzone .le. mzone_puls_bot
     $                      .and. ipulse .eq. 0) then
                             if (rmulti_seas_pred(month).gt.
     $                        rmulti_seas_thres(1,mzone,ioper_opt).and.
     $                        multi_seas_pred_opt.eq.1) then
                              cmulti_seas_forec = cgen_multi_seas_categ(1
     $                            ,mzone,ioper_opt)
                              iplslevel =ipulse_level_multi_seas(1,mzone,
     $                         ioper_opt)
                             endif
                            else
                              cmulti_seas_forec = 'PULSE C'
                              cseas_forec = 'PULSE C'
                            endif
                            call qpulse(rstgpls,RLKSTG,iplslevel
     $                       ,slpulse,calpulse,month,iday,mzone)
                            if (nint(calpulse).eq.0) rleakance = 5.0
                         endif
                        else
                           calpulse = 0.0
                           slpulse  = 0.0
                           rleakance = 5.0
                        endif
                        KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
     $                     /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
     $                     + rleakance

                        if (stagelo .ge. stg_backflw_stl) then
                          KFLO(ifls308)= amax1(slpulse-stlro
     $                     - sledmd_met_by_lok/1.9835,0.0) +eqs308/1.9835
                          REGS308 = amax1(slpulse - stlro
     $                     - sledmd_met_by_lok/1.9835 ,0.0)
                        else
                          KFLO(ifls308)= amax1(slpulse-sledmd_met_by_lok
     $                     /1.9835,0.0) + eqs308/1.9835
                          REGS308 = amax1(slpulse-sledmd_met_by_lok
     $                     /1.9835,0.0)
                        endif
                        REGS77  = amax1(calpulse-c43est_met_by_lok
     $                     /1.9835-cal_rnff_frm_reg,0.0)
                      else
c
c       LOK stage NOT within 0.5 ft ZONE C
c
                        if (multi_seas_pred_opt.eq. 1) then
                         if (mzone .ge. mzone_puls_top .and.
     $                    mzone .le. mzone_puls_bot) then
                          cstg_lok_pres = 'NOT_WITH_.5_FT_ZONEC'
                         endif
                        endif
                        if ( iclimate_opt .eq. 0 .or.
     $                  ( HURRPRED .and. no_of_hurr_pred .gt.
     $                  no_pred_storms_thres)
     $                  .or. (climate_index(month) .ge.
```

```
$                  rmin_clim_indx_thres(mzone,2,ioper_opt) .and.
$                  multi_seas_pred_opt.eq.0) .or. (climate_index
$                  (month) .ge. rmin_clim_indx_thres(mzone,2
$                  ,ioper_opt) .and. rmulti_seas_pred(month) .ge.
$                  rmulti_seas_thres(2,mzone,ioper_opt).and.
$                  multi_seas_pred_opt.eq.1) .or. ipulse.eq.1) then
                    if (multi_seas_pred_opt .ne. 0) then
                     if (ipulse .eq. 0) then
                        if (rmulti_seas_pred(month).gt.
$                         rmulti_seas_thres(3,mzone,ioper_opt)) then
                            iplslevel = ipulse_level_multi_seas(3
$                            ,mzone,ioper_opt)
                        else
                            iplslevel = ipulse_level_multi_seas(2
$                            ,mzone,ioper_opt)
                        endif
                     endif
                     if (mzone .ge. mzone_puls_top .and. mzone
$                     .le. mzone_puls_bot) then
                      if (climate_index(month).ge.
$                      rmin_clim_indx_thres(mzone,2,ioper_opt)
$                      .and. rmulti_seas_pred(month) .ge.
$                      rmulti_seas_thres(2,mzone,ioper_opt)) then
                       cseas_forec = cgen_seas_categ(3,mzone
$                      ,ioper_opt)
                       cmulti_seas_forec = cgen_multi_seas_categ
$                         (2,mzone,ioper_opt)
                      else
                       if (climate_index(month).ge.
$                       rmin_clim_indx_thres(mzone,2,ioper_opt)) then
                        cseas_forec = cgen_seas_categ(3,mzone
$                       ,ioper_opt)
                       else
                        cseas_forec = 'PULSE_C'
                       endif
                       if (rmulti_seas_pred(month) .ge.
$                       rmulti_seas_thres(2,mzone,ioper_opt)) then
                        cmulti_seas_forec = cgen_multi_seas_categ
$                         (2,mzone,ioper_opt)
                       else
                        cmulti_seas_forec = 'PULSE_C'
                       endif
                      endif
                     else
                      cmulti_seas_forec = 'PULSE_C'
                      cseas_forec = 'PULSE_C'
                     endif
                    else
                     if (iclimate_opt.eq.1) then
                      if (mzone .ge. mzone_puls_top .and.
$                      mzone .le. mzone_puls_bot) then
                      if (climate_index(month) .ge.
$                       rmin_clim_indx_thres(mzone,2,ioper_opt))
$                       then
                        cseas_forec = cgen_seas_categ(3,mzone
$                       ,ioper_opt)
                      else
```

```
                        cseas_forec = 'PULSE_C'
                      endif
                     else
                      cseas_forec = 'PULSE_C'
                     endif
                    endif
                  endif
                  if (izone_flex_eaares_opt.eq.0 .or.
$                  (izone_flex_eaares_opt.eq.1 .and.
$                  ((ieaarsn_no.le.ntotres .and.
$                  stg_eaa_res_2a .gt.stgmx(ieaarsn_no)-0.20).or.
$                  (stg_eaa_res_2b.gt.stgmx(ieaarss_no)-0.20
$                  .and.ieaarss_no .le.ntotres)) .or.
$                  ipulse.eq.1)) then
                    call qpulse(rstgpls,RLKSTG,iplslevel
$                    ,slpulse,calpulse,month,iday,mzone)
                    rleakance = 0.0
                  else
                    calpulse = 0.0
                    slpulse  = 0.0
                    rleakance = 5.0
                  endif
                 else
                  if (climate_index(month) .ge.
$                  rmin_clim_indx_thres(mzone,2,ioper_opt)) then
                  cseas_forec =cgen_seas_categ(3,mzone,ioper_opt)
                  else
                  cseas_forec =cgen_seas_categ(2,mzone,ioper_opt)
                  endif
                  if (rmulti_seas_pred(month) .ge.
$                  rmulti_seas_thres(2,mzone,ioper_opt) .and.
$                  multi_seas_pred_opt.eq.1) then
                  if (mzone .le. mzone_puls_bot
$                   .and. ipulse .eq. 0) then
                    cmulti_seas_forec = cgen_multi_seas_categ
$                      (2,mzone,ioper_opt)
                    iplslevel = ipulse_level_multi_seas(1,
$                   mzone,ioper_opt)
                  else
                    cmulti_seas_forec = 'PULSE C'
                    cseas_forec = 'PULSE C'
                  endif
                  else
                  if ((rmulti_seas_pred(month).gt.
$                  rmulti_seas_thres(1,mzone,ioper_opt)
$                  .and.multi_seas_pred_opt.eq.1) .or.
$                  ipulse .eq. 1) then
                    if (multi_seas_pred_opt.eq.1) then
                      if (mzone .le. mzone_puls_bot
$                     .and. ipulse .eq. 0) then
                        if (rmulti_seas_pred(month).gt.
$                       rmulti_seas_thres(1,mzone,ioper_opt))then
                         cmulti_seas_forec=cgen_multi_seas_categ
$                        (1,mzone,ioper_opt)
                         iplslevel = ipulse_level_multi_seas(1,
$                        mzone,ioper_opt)
                        endif
```

```
                          else
                            cmmulti_seas_forec = 'PULSE C'
                            cseas_forec = 'PULSE C'
                          endif
                        else
                          if (mzone .le. mzone_puls_bot
     $                    .and. ipulse .eq. 0) then
                            iplslevel = ipulse_level_in_zone(mzone)
                          else
                            cmmulti_seas_forec = 'PULSE C'
                            cseas_forec = 'PULSE C'
                          endif
                        endif
                       endif
                      endif
                      call qpulse(rstgpls,RLKSTG,iplslevel
     $                ,slpulse,calpulse,month,iday,mzone)
                      if (nint(calpulse).eq.0) rleakance = 5.0
                    endif
                    KFLO(ifls77) = amax1(calpulse-c43est_met_by_lok
     $                    /1.9835-cal_rnff_frm_reg,0.0) +eqs77/1.9835
     $                    + rleakance
                    if (stagelo .ge. stg_backflw_stl) then
                      KFLO(ifls308)= amax1(slpulse-stlro
     $                  - sledmd_met_by_lok/1.9835,0.0)+eqs308/1.9835
                      REGS308 = amax1(slpulse - stlro
     $                  - sledmd_met_by_lok/1.9835 ,0.0)
                    else
                      KFLO(ifls308)= amax1(slpulse-sledmd_met_by_lok
     $                       /1.9835,0.0) + eqs308/1.9835
                      REGS308 = amax1(slpulse -sledmd_met_by_lok
     $                       /1.9835,0.0)
                    endif
                    REGS77  = amax1(calpulse -c43est_met_by_lok
     $                      /1.9835-cal_rnff_frm_reg,0.0)
                  endif
                else
c
c    Tributary Conditions are DRY
c
                  if (itrib_hydro_cond_opt .eq. 1) ctrib_hydro_cond
     $                = cgen_trib_hydro_categ(1,mzone)
                  call qpulse(rstgpls,RLKSTG,iplslevel
     $               ,slpulse,calpulse,month,iday,mzone)
                  if (nint(calpulse) .eq. 0) rleakance = 5.0
                  KFLO(ifls77) = amax1(calpulse- c43est_met_by_lok
     $                /1.9835-cal_rnff_frm_reg,0.0) + eqs77/1.9835
     $                + rleakance
                  if (stagelo .ge. stg_backflw_stl) then
                    KFLO(ifls308)= amax1(slpulse-stlro
     $                - sledmd_met_by_lok/1.9835,0.0) + eqs308/1.9835
                    REGS308 = amax1(slpulse - stlro
     $                - sledmd_met_by_lok/1.9835,0.0)
                  else
                    KFLO(ifls308)= amax1(slpulse-sledmd_met_by_lok
     $                /1.9835,0.0) + eqs308/1.9835
                    REGS308 = amax1(slpulse -sledmd_met_by_lok
```

```
     $                /1.9835,0.0)
                  endif
                  REGS77  = amax1(calpulse-c43est_met_by_lok
     $                /1.9835-cal_rnff_frm_reg,0.0)
               endif
            ELSE
c
c  LOK stages are below schedule for releases to tidewater
c
               if (iclimate_opt .eq. 1) clokzone = 'BEL_SCH'
               KFLO(ifls77) =  eqs77/1.9835 + 5.0
               KFLO(ifls308) = eqs308/1.9835
               rleakance = 5.0
            ENDIF
c
 1000     IF (es_dmnd_acc_freq .ne. 'DLY') then
          if (iday.ne.numd(month)) then
c for Caloosahatchee Estuary
             if (REGS77.eq.0.0) then
                addl_flow = 5.0 * 1.9835
             else
                addl_flow = REGS77 * 1.9835
             endif
             if (addl_flow.gt.0.1) then
c regulatory flows for today could effectively decrease the target
c supplemental LOK release or increase the maximum available backflow
c for the remainder of the month
               if (lok_supp_day_c43est.gt.0. .or.
     &             supp_day_c43est_tmp.gt.0.) then
                  rem_supp = lok_supp_day_c43est*(numd(month)-iday)
     &               + supp_day_c43est_tmp
                  if (addl_flow.gt.rem_supp) then
c no supplemental LOK release is needed for the rest of the month,
c and backflow for the remainder of the month is now possible
                     lok_supp_day_c43est = 0.
                     bflo_cap_c43est = addl_flow - rem_supp
                     reg2est1 = rem_supp/1.9835
                  else
c only a reduction in supplemental LOK release can be realized
                     lok_supp_day_c43est = (rem_supp - addl_flow)
     &                                   / (numd(month)-iday)
                     reg2est1 = addl_flow/1.9835
                  endif
               else
c                   if (bflo_cap_c43est.eq.0.) then
c                      write(*,*)
c     &             'lok_supp_day & bflo_cap for c43 are both zeroes!'
c                     write(*,'(5x,3i5)')ifyr+iyear-1,month,iday
c                     stop
c                  endif
c increase potential ( maximum available) backflow for rest of month
                  bflo_cap_c43est = bflo_cap_c43est + addl_flow
               endif
clmbasr>>
             elseif (supp_day_c43est_tmp.gt.0.) then
c remaining requirement for today only (supp_day_c43est_tmp) has to be
c included in the remainder-of-month daily estuarine supplemental requirement
```

```
                rem_supp = lok_supp_day_c43est*(numd(month)-iday)
     &             + supp_day_c43est_tmp
                lok_supp_day_c43est = rem_supp/(numd(month)-iday)
c note: Adjustment for bflo_cap_c43est already done in resasr_sim subroutine
clmbasr<<
             endif
c
c for St. Lucie Estuary
             addl_flow = REGS308 * 1.9835
             if (addl_flow.gt.0.1) then
c regulatory flows for today could effectively decrease the target
c supplemental LOK release or increase the maximum available backflow
c for the remainder of the month
                if (lok_supp_day_c44est.gt.0. .or.
     &              supp_day_c44est_tmp.gt.0.) then
                   rem_supp = lok_supp_day_c44est*(numd(month)-iday)
     &                + supp_day_c44est_tmp
                   if (addl_flow.gt.rem_supp) then
c no supplemental LOK release is needed for the rest of the month,
c and backflow for the remainder of the month is now possible
                      lok_supp_day_c44est = 0.
                      bflo_cap_c44est = addl_flow - rem_supp
                      reg2est2 = rem_supp/1.9835
                   else
c only a reduction in supplemental LOK release can be realized
                      lok_supp_day_c44est = (rem_supp - addl_flow)
     &                                     / (numd(month)-iday)
                      reg2est2 = addl_flow/1.9835
                   endif
                else
c                   if (bflo_cap_c44est.eq.0.) then
c                      write(*,*)
c     &              'lok_supp_day & bflo_cap for c44 are both zeroes!'
c                     write(*,'(5x,3i5)')ifyr+iyear-1,month,iday
c                     stop
c                  endif
c increase potential ( maximum available) backflow for rest of month
                   bflo_cap_c44est = bflo_cap_c44est + addl_flow
                endif
clmbasr>>
             elseif (supp_day_c44est_tmp.gt.0.) then
c remaining requirement for today only (supp_day_c44est_tmp) has to be
c included in the remainder-of-month daily estuarine supplemental requirement
                rem_supp = lok_supp_day_c44est*(numd(month)-iday)
     &             + supp_day_c44est_tmp
                lok_supp_day_c44est = rem_supp/(numd(month)-iday)
c note: Adjustment for bflo_cap_c44est already done in resasr_sim subroutine
clmbasr<<
             endif
          endif
          ENDIF
clmb<
           eqs80 = eqs80 + REGS308 * 1.9835
           eqs79 = eqs79 + REGS77  * 1.9835
c
          if (iclimate_opt .eq. 1) then
           do k = 1,LEN(ctrib_hydro_cond)
```

```
          if (ctrib_hydro_cond(k:k).ne.' ') then
           j = k
          endif
         enddo
         do k = 1,j
          if (ctrib_hydro_cond(k:k).eq.' ') then
           ctrib_hydro_cond(k:k) = '_'
          endif
         enddo
         do k = 1,LEN(cseas_forec)
          if (cseas_forec(k:k).ne.' ') then
           j = k
          endif
         enddo
         do k = 1,j
          if (cseas_forec(k:k).eq.' ') then
           cseas_forec(k:k) = '_'
          endif
         enddo
         do k = 1,LEN(cmulti_seas_forec)
          if (cmulti_seas_forec(k:k).ne.' ') then
           j = k
          endif
         enddo
         do k = 1,j
          if (cmulti_seas_forec(k:k).eq.' ') then
           cmulti_seas_forec(k:k) = '_'
          endif
         enddo
          write(161,1611) ifyr+iyear-1,month,iday,clokzone
     $    ,ctrib_hydro_cond,cstg_lok_pres,cseas_forec,cmulti_seas_forec
     $    ,REGS77,eqs79/1.9835,REGS308,eqs80/1.9835
        endif
1611    format(I5,2i3,2x,A7,2x,a7,2x,a20,2x,a7,2x,a7,4(3x,f6.0))
        if (iclimate_opt .eq. 1) then
         do k = 1,LEN(ctrib_hydro_cond_wca)
          if (ctrib_hydro_cond_wca(k:k).ne.' ') then
           j = k
          endif
         enddo
         do k = 1,j
          if (ctrib_hydro_cond_wca(k:k).eq.' ') then
           ctrib_hydro_cond_wca(k:k) = '_'
          endif
         enddo
         do k = 1,LEN(cseas_forec_wca)
          if (cseas_forec_wca(k:k).ne.' ') then
           j = k
          endif
         enddo
         do k = 1,j
          if (cseas_forec_wca(k:k).eq.' ') then
           cseas_forec_wca(k:k) = '_'
          endif
         enddo
         do k = 1,LEN(cmulti_seas_forec_wca)
          if (cmulti_seas_forec_wca(k:k).ne.' ') then
```

```
            j = k
           endif
          enddo
          do k = 1,j
           if (cmulti_seas_forec_wca(k:k).eq.' ') then
            cmulti_seas_forec_wca(k:k) = '_'
           endif
          enddo
           write(171,1711) ifyr+iyear-1,month,iday,clokzone_wca
     $     ,ctrib_hydro_cond_wca,cseas_forec_wca,cmulti_seas_forec_wca
     $     ,(ceverg_needs(i),i=1,4),regs354,nnrreg,regs352,hlsbreg
     $     ,(reg_lim_indic(i),i=1,4)
         else
           write(171,1712) ifyr+iyear-1,month,iday,clokzone_wca
     $     ,(ceverg_needs(i),i=1,4),(reg_lim_indic(i),i=1,4)
     $     ,regs354,nnrreg,regs352,hlsbreg
         endif
 1711    format(I5,2i3,2x,A7,2x,a7,2x,a7,2x,a7,2x,4(A3,2x),4f7.0
     $         ,4(2x,A6))
 1712    format(I5,2i3,2x,A7,4(2x,A3),4(2x,A6),2x,4F7.0)
c
         if (REGS77 .eq. 0.0) then
            eqs79 = eqs79 + 5.0 * 1.9835
         endif
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLS77)
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLS308)
C
        if (lok_split_option .eq. 0) then
         LKETVOL = LOKETVOL(STAGELO,MONTH,rf_factor_lok)
         RMETVOL = RMETVOL + LKETVOL
         rainlok1 = rainlok * (466000./area_lok_rain)
         VOLTOLO = -1.9835*(KFLO(IFLS3)+KFLO(IFLS2)+KFLO(IFLHG5)
     &                +KFLO(IFLS77)+KFLO(IFLS308)+KFLO(IFLAGQ)
     &                +KFLO(iflc10a)) + s65ernff(jday)
     &                + s4bsn_to_lok - dmds236 - dmds4 - dmdsemind(jday)
     &                + (rnorth_stor_to_lok - rlok_to_north_stor)*1.9835
     &                + tcnsrnff(jday) + tcrnbsres_to_lok*1.9835
     &                + asr2lok*1.9835 - lok2asr*1.9835
     &                + cresbp2lok - ws_to_roten_frm_lok*1.9835
     &                - fplresws(jday) + chgrnfistapoga(jday)
     &                - chgdmdistapoga(jday) + stl_res_backflw*1.9835
     &                + chgrnfnls(jday) - chgdmdnls(jday)
     &                + chgrnfnels(jday) - chgdmdnels(jday)
     &                + uisrnff(jday)
         FINSTOR  = SPREV - LKETVOL + CFSOTH*1.9835 - rainlok1
     &                + VOLTOLO + rf_factor_lok(month)*rainlok1
     &                - amax1(CFSOTH*1.9835 - rainlok1,0.0)
     &                *(1.0-ro_factor_lok(month))
c        print *,FINSTOR,SPREV,LKETVOL,CFSOTH*1.9835,rainlok1,
c    &   KFLO(IFLS3),KFLO(IFLS2),KFLO(IFLHG5),KFLO(IFLAGQ),
c    &   dmds236,dmds4,KFLO(iflc10a),ws_to_roten_frm_lok,'  LOKB'
        else
         lketvol = loketlitz(stagelo,month,rf_factor_lok)
         rmetvol = rmetvol + lketvol
         lketvol_res = loketres(stagelo_res,month,rf_factor_lok)
```

```
          rmetvol_res = rmetvol_res + lketvol_res
          if (stagelo .gt. stgref .or. stagelo_res .lt. 2.0) then
            vol_flow_litzone = -(KFLO(IFLS3)+KFLO(IFLS2)
     $      + KFLO(IFLHG5)+KFLO(IFLAGQ)+amax1(KFLO(iflc10a),0.0))*1.9835
     $      -(amax1(eqs308,0.0)+eqs77)+s65ernff(jday)+s4bsn_to_lok-DMDS4
     $      - dmdsemind(jday) + tcnsrnff(jday) + tcrnbsres_to_lok*1.9835
     $      - ws_to_roten_frm_lok*1.9835 - fplresws(jday) - DMDS236
     $      + eallokres1 + eallokres2 + c43dmd_met_by_lokres
     $      + c44dmd_met_by_lokres + chgrnfistapoga(jday)
     $      - chgdmdistapoga(jday) + chgrnfnls(jday)
     $      - chgdmdnls(jday) - chgdmdnels(jday) + uisrnff(jday)
          else
            vol_flow_litzone = -eqs77-amax1(eqs308,0.0)
     $      + s65ernff(jday)-amin1(KFLO(IFLS3),0.0)*1.9835
     $      - amin1(KFLO(IFLS2),0.0)*1.9835 + tcrnbsres_to_lok*1.9835
     $      + tcnsrnff(jday) + s4bsn_to_lok - amin1(KFLO(IFLAGQ),0.0)
     $      * 1.9835 + eallokres1 + eallokres2 + c43dmd_met_by_lokres
     $      + c44dmd_met_by_lokres - amin1(DMDS236,0.0)
     $      - (regs354+regs351+regs352)*1.9835
     $      - total_divers_to_eaares*1.9835 + chgrnfistapoga(jday)
     $      + chgrnfnls(jday) + uisrnff(jday)
          endif
          voltolo = vol_flow_litzone + (rnorth_stor_to_lok
     $        - rlok_to_north_stor)*1.9835 + asr2lok*1.9835
     $        - lok2asr*1.9835 + cresbp2lok - regs77*1.9835
     $        - regs308*1.9835 - rleakance*1.9835
          finstor = sprev - lketvol + cfsoth_litz*1.9835
     $     + voltolo - (1.0 - rf_factor_lok(month))*rainlok_litz
     $     -  amax1(cfsoth_litz*1.9835 - rainlok_litz,0.0)
     $     * (1.0 - ro_factor_lok(month))
          finstor_res = sprev_res - lketvol_res + vol_flow_res
     $     + cfsoth_res*1.9835 - (1.0 - rf_factor_lok(month))
     $     * rainlok_res - amax1(cfsoth_res*1.9835 - rainlok_res,0.0)
     $     * (1.0 - ro_factor_lok(month))
          stagelo_res = stg2stores(finstor_res,1)
        endif
c
c  sum monthly rainfall on LOK and MDS adjustmentfor Water Budget
c
        IF (lok_split_option .eq. 0) then
          rmthly_rf_lok = rmthly_rf_lok + (rainlok*12.0
     $    * rf_factor_lok(month)/area_lok_rain)
          rmds_adjust = rainlok* (1.0-rf_factor_lok(month))
     $    /1.9835
          cfsoth1 = cfsoth-rmds_adjust-rainlok1*rf_factor_lok(month)
     $    /1.9835 - amax1(CFSOTH*1.9835- rainlok1,0.0)*(1.0 -
     $    ro_factor_lok(month))/1.9835
        ELSE
          rmthly_rf_lok = rmthly_rf_lok + (rainlok*12.0
     $    *rf_factor_lok(month)/area_lok_rain)*(area_lok_rain/466000.)
          rmds_adjust_litz = rainlok_litz
     $    *(1.0-rf_factor_lok(month))/1.9835
          cfsoth_litz1 = cfsoth_litz - rmds_adjust_litz
     $    - rainlok_litz * rf_factor_lok(month)/1.9835
     $    - amax1(CFSOTH_litz*1.9835- rainlok_litz,0.0)
     $    * (1.0-ro_factor_lok(month))/1.9835
          cfsoth1 = cfsoth_litz1
```

```
          rmds_adjust_res = rainlok_res
     $    *(1.0-rf_factor_lok(month))/1.9835
          cfsoth_res1 = cfsoth_res - rmds_adjust_res
     $    -rainlok_res* rf_factor(month)/1.9835
     $    - amax1(CFSOTH_res*1.9835- rainlok_res,0.0)
     $    * (1.0-ro_factor_lok(month))/1.9835
        ENDIF
c
c  output components of LOK ET
c
        if (lp(10)) then
          write(181,'(i5,2i4,3x,f6.2,4f8.0)') ifyr+iyear-1,month,iday
     $    ,stagelo,OPNETVOL,LITETVOL,DRYETVOL,LKETVOL
        endif
c
c  route water supply from LOK to STAs
c
       DO  IA = 1,NMAWS
       iresnum = indxaws(ia)
          IF (sfactor(iresnum) .GE. sfactmin .OR. ires_small_sim
     $    (iresnum) .eq. 'NO ') THEN
            IB = ibsn_no_res(iresnum)
            ws_deliv_to_sta = WSDEL_TO_STA(INDXAWS(IA))
            DO  IY = 54,42,-1
              DO  NODE = ISUM(IY)+1,ISUM(IY+1)
                IF (IBSN(NODE) .EQ. IB) THEN
                  dpth_below_min = ells(node)+DPHWS(INDXAWS(IA))-amax1(
     $             stage_res(node,1),ells(node)) + (ells(node) -
     $             amin1(ells(node),stage_res(node,1)))*s(node)
                  dpth_below_min = amax1(dpth_below_min,0.0)
                  POND(NODE) = POND(NODE) + amin1(dpth_below_min
     $            ,ws_deliv_to_sta*86400./(GDAR*(1.0-carfact(NODE))))
                  ws_deliv_to_sta = ws_deliv_to_sta -
     $            amin1(dpth_below_min*GDAR/86400.,ws_deliv_to_sta)
                ENDIF
              ENDDO
            ENDDO
          ELSE
            DO IY = 54,42,-1
              DO NODE = ISUM(IY)+1,ISUM(IY+1)
                 DO ir = 1,noresincell(NODE)
                    IF (iresnum .EQ. ires_index_in_cell(NODE,ir))
     $              THEN
                      pond_in_res(node,ir) = pond_in_res(NODE,ir)
     $                + WSDEL_TO_STA(INDXAWS(IA)) * 86400.
     $                / (NNODES(INDXAWS(IA)) * sfactor(indxaws(ia))
     $                * GDAR)
                    ENDIF
                 ENDDO
               ENDDO
            ENDDO
            ws_deliv_to_sta = 0.0
          ENDIF
          QSTAIN(INDXAWS(IA)) = QSTAIN(INDXAWS(IA))
     $        + WSDEL_TO_STA(INDXAWS(IA))-ws_deliv_to_sta
          VOLWTR(INDXAWS(IA)) = VOLWTR(INDXAWS(IA))
     $        - WSDEL_TO_STA(INDXAWS(IA))+ws_deliv_to_sta
```

```
          twsuply = twsuply + WSDEL_TO_STA(INDXAWS(IA))
     $        - ws_deliv_to_sta
c
          if(resname(indxaws(ia)) .eq. name_res_spec_code(3)) then
            ws_to_sta5 = wsdel_to_sta(indxaws(ia))-ws_deliv_to_sta
            finstor = finstor - ws_to_sta5 * 1.9835
            dellec = dellec + ws_to_sta5 * 1.9835
          elseif (resname(indxaws(ia)) .eq. name_res_spec_code(19)) then
            ws_to_sta5e = wsdel_to_sta(indxaws(ia))-ws_deliv_to_sta
            ws_to_sta5 = ws_to_sta5 + ws_to_sta5e
            dellec = dellec + ws_to_sta5e * 1.9835
            finstor = finstor - ws_to_sta5e * 1.9835
          elseif(resname(indxaws(ia)) .eq. name_res_spec_code(4)) then
            ws_to_sta6 = wsdel_to_sta(indxaws(ia))-ws_deliv_to_sta
            finstor = finstor - ws_to_sta6 * 1.9835
            dellec = dellec + ws_to_sta6 * 1.9835
          endif
c
       ENDDO
       if (lok_split_option .eq. 0) then
         stagelo = stg2sto(finstor,1)
       else
         stagelo = stg2stolitz(finstor,1)
       endif
c
c  put excess water from LOK to LOKRES
c
       if (stagelo .gt. stgref .and. lok_split_option
     $      .eq. 1) then
         outflow_to_lokres = capac_struc_to_lokres
         finstor = finstor - outflow_to_lokres*1.9835
         stagelo = stg2stolitz(finstor,1)
         finstor_res = finstor_res + outflow_to_lokres*1.9835
         stagelo_res = stg2stores(finstor_res,1)
       else
         outflow_to_lokres = 0.0
       endif
c
c     write(*,'(13f10.0,3i5)')-1.9835*KFLO(IFLS3),-1.9835*KFLO(IFLS2)
c    &            ,-1.9835*KFLO(IFLHG5),-1.9835*KFLO(IFLS77)
c    &            ,-1.9835*KFLO(IFLS308),-1.9835*KFLO(IFLAGQ)
c    &            ,-1.9835*KFLO(iflc10a),-1.9835*TWSUPLY
c    &            , s65ernff(jday),s4bsn_to_lok
c    &            , -dmds236,-dmds4,-dmdsemind(jday)
c    &            ,MONTH,IDAY,IFYR+IYEAR-1
c          print *,FINSTOR,SPREV,FINSTOR-SPREV,LKETVOL,CFSOTH*1.9835,
c    $      VOLTOLO,MONTH,IDAY,IFYR+IYEAR-1,'  END OF LAKEWCA'
      RETURN
      END
```

lakwca.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                    -
C     SCCS ID:  %W% %G% SFWMD Planning Department                    -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: lakwca.inc,v 1.4 2003/07/03 20:08:00 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lakwca.inc,v $
C       $Name: rel-5-5 $                                                    -
C                                                                    -
C                                                                    -
C-----------------------------------------------------
      common /LAKWCA/ stage_import_wca(max_n_targ_area),
     &LRGLO(max_n_targ_area),LHL,IE1,IE2,IE3,L30,LWSLO(max_n_targ_area),
     &IG3A28,IG3B29,IC13,IS37B,INNRC,
     &RCPFACTEAA(max_n_eaa_condt),rcpfacteaa_res(max_n_eaa_condt),
     &IRGPRTY(max_n_eaa_condt),STAGE3A,STAGE3B,STGCA1,SPREV,IES,IFLS235,
     &NSTRCA_REG,STGCA2A,NSTRCA,IG3A3,IG3A4,
     &pct_des(max_n_eaa_condt,max_n_seas),
     &irg_to_res_prty(max_n_eaa_condt),ISTAOPT(max_n_div_to_ecp),
     &STAGELO,RLKET,CFSOTH,TWSUPLY,WSS354,sem_cyp_dmnd(366),WSS351,
     &WSS352,EXCNNR,HILLEXC,ZONECM(12),DAREFET,flow_s8new_to_sta,
     &RMKVOLSTA1,RMKVOLSTA2,RMKVOLSTA3,RMKVOLSTA4,RMETVOL,IESE,lok2asr,
     &STG3A28,STG3A3,STG3A4,IPLSDAY,IPULSE,IFLS4,IFLS65E,
     &LWSLO_floor(max_n_wcas),GRF3AMX,GRF2AMX,FLLKS6,FLLKWPB,
     &nmpts_wca(max_n_targ_area),REGS77,n_fc_zones,
     &rimport_stage(max_n_targ_area,max_n_loc_area),
     &node_import(max_n_targ_area,max_n_loc_area),
     &daily_stg_import_wca(max_n_targ_area),
     &ngrid_cells_daily(max_n_targ_area),REGS308,IACMDR,
     &node_import_daily(max_n_targ_area,max_n_loc_area),
     &avg_import_stage(max_n_targ_area),nadd_imp_area,s5a2_so,s5a2_no,
     &s5a3_no,s5a3_so,icnl_dn_s31,cresbp2lok,s5a_to_sta,
     &dmnd_miami_basin,dmnd_nnr,dmnd_hill,flow_to_import,ista34,
     &flow_to_import_miami,flow_to_import_nnrc,flow_to_import_hill,
     &miamic_cap_upper_grav,miamic_cap_lower_grav,miamic_cap_upper_pump,
     &miamic_cap_lower_pump,nnrc_cap_upper_grav,nnrc_cap_lower_grav,
     &nnrc_cap_upper_pump,nnrc_cap_lower_pump,s354_tw,s351_tw,asr2lok,
     &s352_tw,s8_new_hw,s8_new_tw,s7_new_hw,s7_new_tw,flow_s7new_to_sta,
     &s8_spill_hw,s8_spill_tw,s7_spill_hw,s7_spill_tw,s6_hw,s5a_hw,
```

```
&rcpfacteaa_canal(max_n_eaa_condt),LFLWTRG(max_n_targ_area),
&frc_flow_targ(max_n_str_w_ds_targ),iopt_coth_wcas(2),
&struc_name_w_targ(max_n_str_w_ds_targ),
&rem_inflow_to_meet_holy_sched,iopt_for_reg_releases_s334,
&stl_res_backflw,LWSLO_eaares(max_n_targ_area),
&rmax_lift(max_n_eaa_condt),rmax_tw_eaad(max_n_eaa_condt),
&rmin_lok_stg_forw_pump(max_n_eaa_condt),n_days_week_del_eaa(2),
&idays_wk_del_eaa_ssm_indx(7,2)
 LOGICAL LWSLO,LWSLO_floor,LRGLO,LFLWTRG,LWSLO_eaares
 real miamic_cap_upper_grav,miamic_cap_lower_grav,lok2asr
$,miamic_cap_upper_pump,miamic_cap_lower_pump,nnrc_cap_upper_grav
$,nnrc_cap_lower_grav,nnrc_cap_upper_pump,nnrc_cap_lower_pump
 character struc_name_w_targ*6
 DOUBLE PRECISION stage_import_wca
```

larger_reserv_stor.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: larger_reserv_stor.F,v 1.11 2005/06/10 13:29:11 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/larger_reserv_stor.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE larger_reserv_stor (ia, volwtr, sumrain, qoutmx,
     & total_afact, deltastor, prev_stage_res, tvolwtrl,iseas)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'resadj.inc'

      DIMENSION deltastor(max_n_res),
     & prev_stage_res(max_ncells,max_n_res_in_cell),
     & qoutmx(max_n_res,max_nostr_res), sumrain(max_ncells),
     & total_afact(max_ncells), tvolwtrl(max_n_res), volwtr(max_n_res)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   DETERMINE THE MAXIMUM AVAILABLE STORAGE FOR EACH RESERVOIR

        volwtr(ia)=0.0
```

```
         deltastor(ia)=0.0
         DO ij=1,nasr_from_res(ia)
           avail_vol_to_asr(ia,ij)=0.0
         END DO
         mean_ponding_depth(ia)=0.0
         DO 10 iy=maxy,1,-1
           DO 10 ix=minx(iy),maxx(iy)
           node=ix-minx(iy)+1+isum(iy)
             DO 10 ir = 1,noresincell(node)
                IF (ia .eq. ires_index_in_cell(node,ir)) THEN
                   stage=h(node)+pond(node)
                   IF (stage.le.ells(node)) THEN
                      afact=1.0
                   ELSE
                      afact=total_afact(node)
                   END IF
                   adjust_for_ovlf=sum_chg_dpth_ovl_flw(node)
                   term1_for_adjust_area=(sumet_outside_res(node)
     &             +seep_loss_res(node)-sumrain(node)/12.0
     &             -sum_chg_depth_levee_spg(node))
     &             *((1.0-afact)/afact)
                   term2_for_adjust_area=(outf_fc_frm_res_to_cell(node)+
     &             outf_seep_loc_to_res_cell(node)+adjust_for_ovlf)/afact
                   IF (stage.ge.ells(node)) THEN
                      depsw=(stage-ells(node))/afact +
     &                term1_for_adjust_area-(outf_fc_frm_res_to_cell
     &                (node)+outf_seep_loc_to_res_cell(node))/afact-
     &                sum_chg_dpth_ovl_flw(node)/afact-adjust_dur_excess
     &                (node)
                      depsw=depsw-adjust_for_area_m1(node)
                      stage_res(node,ir)=ells(node)+depsw
                      mean_ponding_depth(ia)=mean_ponding_depth(ia)+depsw
     &                /nnodes(ia)
                      volwtr(ia)=volwtr(ia)+(stgmx(ia)-stage_res(node,
     &                ir))*sfactor(ia)*gdar/86400.
                   ELSE
                      volwtr(ia)=volwtr(ia)+amax1(stgmx(ia)-ells(node),
     &                0.0)*sfactor(ia)*gdar/86400.+(amin1(stgmx(ia),
     &                ells(node))-stage)*s(node)*gdar/86400.
                      stage_res(node,ir)=h(node)+pond(node)
                   END IF
            if (resname(ia).eq.'STA1E ') THEN
            deltastor(ia) = deltastor(ia) + (amax1(stage_res(node,ir)
     &      ,ells(node)) - amax1(prev_stage_res(node,ir)
     &      ,ells(node)) + (amin1(stage_res(node,ir)
     &      ,ells(node)) - amin1(prev_stage_res(node,ir)
     &      ,ells(node)))*s(node)) * sfactor(ia)*gdar/43560.
            endif
c           if (resname(ia).eq. 'STA1W ')
c    $      print *, STAGE,SFACTOR(IA),IA,STGMX(IA),SUMRAIN(NODE)
c    $ ,IR,SUMET(NODE),VOLWTR(IA),MONTH,IDAY,IFYR+IYEAR-1,'  IN STASTOR'
c    $      ,DEPSW,adjust_for_area_m1(node),sum_chg_depth_levee_spg(NODE)
c    $      ,sum_chg_depth_levee_spg(node),adjust_dur_excess(node)
c    $      ,outf_fc_frm_res_to_cell(NODE),POND(NODE)
                ENDIF
   10    CONTINUE
!
```

```
        IF (resname(ia).eq.name_res_spec_code(12)) THEN
          volwtr(ia)=volwtr(ia)+(kflo(ifls335)-kflo(ifls337))
        END IF
C
        DO i=1,noutstr(ia)
          IF (res_out_type(ia,i).ne.'WSPLY') THEN
            DO ip=1,npumptores
              IF (iwmads(ia,i).eq.ibsnres(ip)) THEN
                qoutmx(ia,i)=pumpcap_res(ip)
              END IF
            END DO
            IF (lrestme(ia)) THEN
              qoutmx(ia,i)=(rlength(ia)/restime(ia))*amax1(stgmx(ia)-
     &         ells(nodewa(ia)),0.0)*rwidth(ia)*5280.*5280./86400.
            ELSE
c             IF (wrexp(ia,i,2).eq.1.0) THEN
                IF (inodopt(ia,i,3).eq.1) THEN
                  IF (iwmads(ia,i).gt.ntotres.or.h(nodepw(ia,i,1))+
     &             pond(nodepw(ia,i,1)).le.ells(nodepw(ia,i,1))) THEN
                    afact=1.0
                  ELSE
                    afact=total_afact(nodepw(ia,i,1))
                  END IF
                  adjust_for_ovlf=sum_chg_dpth_ovl_flw(nodepw(ia,i,1))
                  term1_for_adjust_area=(sumet_outside_res(nodepw(ia,i,1
     &            ))+seep_loss_res(nodepw(ia,i,1))-sumrain(nodepw(ia,i,1
     &            )) /12.0-sum_chg_depth_levee_spg(nodepw(ia,i,1)))
     &             *((1.0-afact)/afact)
                  term2_for_adjust_area=
     &             (outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &             outf_seep_loc_to_res_cell(nodepw(ia,i,1))+
     &             adjust_for_ovlf)/afact
                  deptgw=(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))*
     &             s(nodepw(ia,i,1))
                  IF (pond(nodepw(ia,i,1)).gt.deptgw) THEN
                    tail=pond(nodepw(ia,i,1))-deptgw+ells(nodepw(ia,i,1)
     &               )
                  ELSE
                    tail=(pond(nodepw(ia,i,1))-deptgw)/s(nodepw(ia,i,1))
     &               +ells(nodepw(ia,i,1))
                  END IF
                  IF ((tail.ge.ells(nodepw(ia,i,1))
     &             .or.(stage_res(nodepw(ia,i,1),ir).ge.ells(nodepw(ia
     &             ,i,1)).and.iwmads(ia,i).le.ntotres)).and.nodepw(ia,i
     &             ,1).ne.idnstrm(ia,i)) THEN
                    tail=ells(nodepw(ia,i,1))+pond(nodepw(ia,i,1))/
     &               afact+term1_for_adjust_area-
     &               adjust_dur_excess(nodepw(ia,i,1))-
     &               (outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &               outf_seep_loc_to_res_cell(nodepw(ia,i,1)))/afact-
     &               sum_chg_dpth_ovl_flw(nodepw(ia,i,1))/afact+
     &               daily_chg_dpth_ovl_vol1(nodepw(ia,i,1))/afact
!
                    tail=tail-adjust_for_area_m1(nodepw(ia,i,1))
                  END IF
                  IF (sfactor(ia) .GT. sfactmin .OR. ires_small_sim(ia)
     &             .EQ. 'NO ') THEN
```

```
                    IF (tail.gt.ells(nodepw(ia,i,1)).and.nodepw(ia,i,1)
     &               .eq.idnstrm(ia,i)) THEN
                      tail=est_cell_stg_o_res(nodepw(ia,i,1))
                    END IF
                  ENDIF
                  IF (nodepw(ia,i,1).ne.idnstrm(ia,i)) tail=amax1(tail,
     &             ells(nodepw(ia,i,1)))
                ELSE
                  IF (ircnl(ia,i,1).le.nch) THEN
                    IF (nodcr(ircnl(ia,i,1)) .gt. 1) then
                     diff_canlr_stg_frm_ds_end=float(nodcr(ircnl(ia,i,1)
     &               )-NODEPOS_CANLR(ia,i,1))/float(nodcr(ircnl(ia,i,1))
     &               -1)*hdc(ircnl(ia,i,1),iseas)
                    ELSE
                     diff_canlr_stg_frm_ds_end = 0.0
                    ENDIF
                    tail=chdep(ircnl(ia,i,1))+diff_canlr_stg_frm_ds_end
                  ELSE
                    tail=crstelev(ia,i,2)
                  END IF
                END IF
                IF (wrexp(ia,i,2).eq.1.0) THEN
                   qoutmx(ia,i)=weircff(ia,i,2)*(stgmx(ia)-tail)
                ENDIF
c             if (resname(ia).eq.'S332BR') print *,WEIRCFF(IA,I,2)
c    $            ,STGMX(IA),TAIL,QOUTMX(IA,I)
c    $            ,'  IN STASTOR'
c             ELSE
                IF (type(ia,i).eq.'GRAVITY') THEN
                  IF (wrexp(ia,i,2) .NE. 1.0) THEN
                    wcrsth=amax1(crstelev(ia,i,2),tail)
                  ELSE
                    wcrsth=tail
                  ENDIF
                ELSE
                  wcrsth=crstelev(ia,i,2)
                END IF
                IF (wcrsth.gt.crstelev(ia,i,2)) THEN
                  crelev = crstelev(ia,i,2)
                  IF (stgmx(ia) .gt. crelev) THEN
                   submerg_fact = (wcrsth - crelev)/(stgmx(ia) - crelev)
                   rel_crst_length = (stgmx(ia) - crelev)/(stgmx(ia)
     &             - crelev + 1.0)
                   rmod_limit = 0.85 - 0.50*rel_crst_length
                   reduc_fact = (submerg_fact -rmod_limit)
     &             /(1.0 -rmod_limit)
                   reduc_fact = amax1(reduc_fact,0.0)
                   reduc_fact = (1.0 - reduc_fact) ** 0.143
                   qfactor=reduc_fact
                  ELSE
                   qfactor=1.0
                  ENDIF
                ELSE
                  qfactor=1.0
                END IF
                IF (stgmx(ia).gt.crstelev(ia,i,2)) THEN
                  qoutmx(ia,i)=weircff(ia,i,2)*(stgmx(ia)-crstelev(ia,i,
```

```
     &             2))**wrexp(ia,i,2)*qfactor
                 ELSE
                   qoutmx(ia,i)=0.0
                 END IF
                 IF (iwmads(ia,i).lt. ntotres) then
                    qoutmx(ia,i) = amin1(qoutmx(ia,i)
     &              ,tvolwtrl(iwmads(ia,i)))
                    tvolwtrl(iwmads(ia,i)) = tvolwtrl(iwmads(ia,i))
     &              - qoutmx(ia,i)
                 ENDIF
c                if (resname(ia).eq.'STA5W ') print *,qoutmx(ia,i),
c    &           weircff(ia,i,2),stgmx(ia),crstelev(ia,i,2),'  QOUTMX  '
c    &           ,qfactor,crstelev(ia,i,2)
c              END IF
               IF (nodepw(ia,i,1).eq.idnstrm(ia,i).and.inodopt(ia,i,2)
     &          .eq.1) THEN
                 daily_outf_from_res_to_cell_max(idnstrm(ia,i))=
     &            qoutmx(ia,i)*86400./gdar
               END IF
               IF (inodopt(ia,i,3).ne.1.and.ircnl(ia,i,1).le.nch) THEN
                 IF (itrgcnl(ia,i,1) .le. nch) THEN
                    if (nodcr(itrgcnl(ia,i,1)) .gt. 1) then
                      diff_stg_cnl_trg = float((nodcr(itrgcnl(ia,i,1))-
     &                nodepos_rcnl_trg(ia,i,1)))/float(nodcr(itrgcnl
     &                (ia,i,1))-1)*hdc(itrgcnl(ia,i,1),iseas)
                    else
                      diff_stg_cnl_trg = 0.0
                    endif
                    dn_cnl_stg_trg = chdep(itrgcnl(ia,i,1))+
     &              diff_stg_cnl_trg
                 ELSE
                   if (icell_trg_loc(ia,i,1) .gt. 0) THEN
                      dn_cnl_stg_trg = h(icell_trg_loc(ia,i,1))
     &                + pond(icell_trg_loc(ia,i,1))
                   else
                      dn_cnl_stg_trg = -901.0
                   endif
                 ENDIF
                 IF (dn_cnl_stg_trg.lt.dwnstgmx(ia,i,1)+
     &            offset_reg_res(ia,i,1).and.dn_cnl_stg_trg
     &            .ge.dwnstgmx(ia,i,1)) THEN
                    qoutmx(ia,i)=1.0/offset_reg_res(ia,i,1)*(dwnstgmx
     &              (ia,i,1)+offset_reg_res(ia,i,1)-dn_cnl_stg_trg)
     &              *qoutmx(ia,i)
                 ELSE IF (dn_cnl_stg_trg.ge.dwnstgmx(ia,i,1)+
     &             offset_reg_res(ia,i,1)) THEN
                    qoutmx(ia,i)=0.0
                 END IF
               END IF
             END IF
           ELSE
             qoutmx(ia,i)=0.0
           END IF
           volwtr(ia)=volwtr(ia)+qoutmx(ia,i)
         END DO
         rmax_outflow_asr=0.0
         IF (nasr_from_res(ia).gt.0) THEN
```

```
          DO j=1,nasr_from_res(ia)
            rmax_outflow_asr=rmax_outflow_asr+asr_incap(asrnum(2,ia,j,1)
     &       )*1.532
          END DO
        END IF
        volwtr(ia)=volwtr(ia)+rmax_outflow_asr
!
      RETURN
      END
```

lec_et.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: wwilcox $
C     $Id: lec_et.inc,v 1.6 2003/09/05 15:28:28 wwilcox Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et.inc,v $
C       $Name: rel-5-5 $
-
C
C
C-----------------------------------------------------
c Include file for the LEC ET routine variables

      integer nrows, num_lec_nodes, num_use_type, use_lec_et
      integer et_in_def, net_irr_def
      integer et_comp_def, supply_def
      integer shortage_def, est_et_def, est_unres_et_def
      integer et_in_unit, net_irr_unit
      integer et_comp_unit, supply_unit
      integer est_unres_et_unit
      integer shortage_unit, est_et_unit
      integer isum_lec(max_n_rows), max_x_lec(max_n_rows),
     &min_x_lec(max_n_rows)
      integer lec_node_in_model(max_ncells),
     &model_node_in_lec(max_n_cells_lec)

      real smmax(max_ncells), swscap(max_ncells),
     &erzd(max_ncells), rzd(max_ncells)
c     smmax(max_ncells)  = (acre-ft)
c     swscap(max_ncells) = (ft)
c     erzd(max_ncells) = (ft)
c     rzd(max_ncells) = (ft)
      real flirpws(max_ncells),fgirtww(max_ncells),
     &     flirtww(max_ncells)
      real irrig_area(max_ncells,max_n_type)
      real frac_irr_area(max_ncells,max_n_type)
      real frac_irr_area_t(max_ncells)
      real et_poten(max_ncells)
      real et_ponding(max_ncells)
```

```
 real et_sat(max_ncells)
 real infilt(max_ncells), perc(max_ncells)

 real et_unsat(max_ncells), et_unsat_actual(max_ncells),
*     et_unsat_irr(max_ncells), et_unsat_irr_actual(max_ncells),
*     et_unsat_nonirr(max_ncells),
*     et_unsat_nonirr_actual(max_ncells)
*    ,et_unsat_unacct(max_ncells),
*     et_unsat_irr_actual_month(max_ncells)

 real etiu_reduction(max_ncells,max_n_type),
*     etiu_reduction_t(max_ncells),
*     etiu_unrestr(max_ncells,max_n_type),
*     etiu_restr(max_ncells,max_n_type),
*     et_unsat_adj(max_ncells),et_unsat_irr_adj(max_ncells)

 real net_irr_t(max_ncells), supply_t(max_ncells)
 real shortage_t(max_ncells)
 real net_irr(max_ncells, max_n_type)
 real supply(max_ncells,max_n_type)
 real shortage(max_ncells,max_n_type)

 real et_poten_month(max_ncells)
 real et_ponding_month(max_ncells)
 real infilt_month(max_ncells), perc_month(max_ncells)
 real et_unsat_month(max_ncells),
*     et_unsat_irr_month(max_ncells),
*     et_unsat_nonirr_month(max_ncells),
*     et_sat_month(max_ncells)
*    ,et_unsat_unacct_month(max_ncells)
 real net_irr_t_month(max_ncells), supply_t_month(max_ncells)
 real shortage_t_month(max_ncells)
 real net_irr_month(max_ncells,max_n_type)
 real supply_month(max_ncells,max_n_type)
 real shortage_month(max_ncells,max_n_type)
 real shortage_month_pws(max_ncells),
*     demand_month_pws(max_ncells), supply_month_pws(max_ncells),
&     pump_def(max_ncells,12)

 real min_maxappl_rate(max_ncells), nirrmax(max_ncells,max_n_type)

 real x_size, y_size
 real et_unsat_lec(max_n_cells_lec), et_poten_lec(max_n_cells_lec)
 real net_irr_lec(max_n_cells_lec)

 character*80 snap_tag
 character*70 title_et_in
 character*70 title_net_irr
 character*200 net_irr_file, supply_file
 character*200 et_in_file, et_comp_file
 character*200 shortage_file, est_et_file, est_unres_et_file
 character*3 etiu_unrest_print_opt

 common /lec_defns/et_in_def, net_irr_def,
*   et_in_unit, net_irr_unit, est_unres_et_def,
*   nrows, num_lec_nodes, isum_lec, max_x_lec, min_x_lec,
*   x_size, y_size, et_unsat_lec, et_poten_lec, net_irr_lec,
```

```
*   lec_node_in_model, model_node_in_lec, num_use_type,
*   use_lec_et, et_comp_def, supply_def, et_comp_unit,
*   supply_unit, shortage_unit, est_et_unit,
*   shortage_def, est_et_def ,est_unres_et_unit

 common /lec_et_data/et_poten, et_ponding, et_sat, et_unsat_unacct,
*   et_unsat, et_unsat_actual, et_unsat_irr, et_unsat_irr_actual,
*   et_unsat_nonirr, et_unsat_nonirr_actual, infilt, perc,
*   net_irr,  net_irr_t, supply, supply_t, shortage, shortage_t,
*   et_poten_month, et_ponding_month, infilt_month, perc_month,
*   et_unsat_month, et_unsat_irr_month, et_unsat_unacct_month,
*   et_unsat_nonirr_month, et_sat_month, et_unsat_irr_actual_month,
*   net_irr_month, net_irr_t_month, supply_month, supply_t_month,
*   shortage_month, shortage_t_month, shortage_month_pws,
*   irrig_area, frac_irr_area, frac_irr_area_t,flirpws, pump_def,
*   smmax, swscap, erzd, rzd, demand_month_pws, supply_month_pws,
*   etiu_reduction,etiu_reduction_t,et_unsat_adj,et_unsat_irr_adj,
*   etiu_unrestr, etiu_restr, min_maxappl_rate, nirrmax, fgirtww,
*   flirtww

 common /char_vars/ snap_tag, title_et_in, title_net_irr,
*   et_in_file, net_irr_file, et_comp_file, supply_file,
*   shortage_file, est_et_file, etiu_unrest_print_opt,
*   est_unres_et_file
```

lec_et_comp.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lec_et_comp.F,v 1.6 2003/07/03 20:07:31 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et_comp.F,v $
C   $Name: rel-5-5 $
C
C
C  Subroutine computes the ET in nodes that are in the LEC
C  developed area.  This option is used when the variable USE_LEC_ET > 0.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_et_comp (node, ponding, infiltration, recharge,
     & land_use, head, depth_drz, depth_srz, elev_land, basin_no,
     & et_zone, ag_coeff, month, jday, iday, total_small_afact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'

      INTEGER basin_no, et_zone, jday, land_use, month, node

      REAL ag_coeff, depth_drz, depth_srz, elev_land, gw_depth, head,
     & infiltration, k, kc, kc7(12), kc8(12), kc9(12), ponding,
     & recharge, temp_et_unsat,total_small_afact(max_ncells)

      DATA kc7 /0.64,0.69,0.87,0.95,0.86,0.66,0.61,0.66,0.71,0. 87,
     & 0.93,0.88/
      DATA kc8 /0.80,0.60,0.55,0.80,0.95,1.00,1.05,1.05,1.05,1. 00,
     & 0.95,0.90/
```

```
      DATA kc9 /0.70,0.75,0.81,0.95,0.95,0.98,0.98,0.98,0.94,0. 85,
     & 0.75,0.70/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

c FIRST, ACCOUNT FOR PREPROCESSED ET FOR THE UNSATURATED ZONE

      IF (erzd(node) .EQ. 0.)THEN
         whc = 0.0
      ELSE
         whc = smmax(node) / ( (GDAR*(1.0-carfact(node)
     &   -total_small_afact(node))/43560.) * erzd(node) )
      ENDIF
c determine water holding capacity and current root zone depth
      IF (head .GE. elev_land) THEN
         rzd(node) = 0.0
         swscap(node) = 0.0
      ELSE
         rzd(node) = amin1(erzd(node),elev_land-head)
         swscap(node) = whc * rzd(node)
      ENDIF
c Set et_unsat to temporary variable temp_et_unsat. The temp variable
c will be changed while the original variable remains unchanged.
c next 3 lines for debugging only
cc       et_unsat(node) = 0.0
cc       et_unsat_irr(node) = 0.0
cc       et_unsat_nonirr(node) = 0.0
      temp_et_unsat = et_unsat_adj(node)
c note: et_unsat = et_unsat_irr + et_unsat_nonirr
c update solmx()
      solmx(node) = solmx(node) + infiltration
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c Enter this section IF ETU > 0, account for ETU from updated
c moisture in the unsaturated zone
      IF (temp_et_unsat .GT. 0.) THEN
         IF (solmx(node) .LE. temp_et_unsat) THEN
c problem here:
c entire etu cannot be accounted from updated moisture (infilt and uz moisture)
c content only, the other 'source' is an artifact of sfwmm/afsirs inconsistency
c This needs to be checked out once REAL data are available.
c what could be accounted for is the current moisture in the unsaturated zone
            et_unsat_actual(node) = solmx(node)
            temp_et_unsat = temp_et_unsat - solmx(node)
            solmx(node) = 0.
            DO ijk=65,1,-1
               IF (isum(ijk) .LT. node)THEN
                  irownum = ijk
                  icolnum = node - isum(ijk) + minx(ijk) - 1
                  GO TO 10
               ENDIF
            END DO
   10       CONTINUE
clb10              WRITE(*,1) node, icolnum, irownum, temp_et_unsat,
clb     &                     iyear, month, jday
clb                pause
         ELSE
            et_unsat_actual(node) = et_unsat_adj(node)
```

```
            temp_et_unsat = 0.
c entire etu can be accounted from infilt and uz moisture content
            solmx(node) = solmx(node) - et_unsat_actual(node)
         ENDIF
      ENDIF
c
c temp_et_unsat should be non-negative at this point
      et_unsat_unacct(node) = temp_et_unsat
c
c Calculate et_unsat_irr_actual() and et_unsat_nonirr_actual()
c
      IF ( et_unsat_actual(node) .GE. et_unsat_irr_adj(node) )THEN
c et_unsat_irr_adj() is fully accounted for
         et_unsat_irr_actual(node) = et_unsat_irr_adj(node)
         et_unsat_nonirr_actual(node) = et_unsat_actual(node) -
     $   et_unsat_irr_actual(node)
      ELSE
c et_unsat_irr_adj() has priority over et_unsat_nonirr()
         et_unsat_irr_actual(node) = et_unsat_actual(node)
         et_unsat_nonirr_actual(node) = 0.0
      ENDIF
c
c check IF ENDing solmx() exceeds swscap()
      IF (solmx(node) .LT.swscap(node)) THEN
c field capacity is not exceeded and solmx() remains the same
c while percolation is zero
         perc(node) = 0.0
      ELSE
c field capacity is exceeded and solmx is reduced to field capacity
c and percolation is the excess moisture
         perc(node) = solmx(node) - swscap(node)
         recharge = recharge + perc(node)
         solmx(node) = swscap(node)
         perc(node) = perc(node) * (1.0 - carfact(node))
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
c SECOND, COMPUTE ET FROM THE SATURATED ZONE
c
C THE VALUE OF KMAX REPRESENTS THAT OF AN OPEN WATER SURFACE.
C ALL LAND USES ARE ASSUMED TO BE OPEN WATER WHEN THE PONDING
C DEPTH IS GREATER THAN DMAX(ARBITRARY). AS PONDING GETS
C GREATER THIS COULD MEAN THAT THE PAN COEFFICIENT GROWS OR
C DECREASES.
C
      K = ETK(land_use , JDAY)
C
      IF ((basin_no .GE. 7 .AND. basin_no .LE. 10 .AND. land_use.NE.6)
     $ .OR. basin_no .EQ. 40) THEN
         FAC = ag_coeff
      ELSE
         FAC = 1.0
      ENDIF
C
      IF (land_use .EQ. 7) THEN
         KC = KC7(MONTH) * FAC * et_cal_adj_fact(land_use,MONTH)
      ELSEIF(land_use .EQ. 8) THEN
```

```
         KC = KC8(MONTH) * FAC * et_cal_adj_fact(land_use,MONTH)
      ELSEIF(land_use .EQ. 9) THEN
         KC = KC9(MONTH) * FAC * et_cal_adj_fact(land_use,MONTH)
      ELSE
         KC = 1.0 * et_cal_adj_fact(land_use,MONTH)
      ENDIF
C
C
      gw_depth = elev_land - (head + ponding)
c
      et_sat_max = (et_poten(node) - et_unsat_adj(node))*12.
C
      IF (gw_depth .GT. 0.0) THEN
C
C LAND IS NOT FLOODED
         IF (gw_depth .GE. depth_drz) THEN
c  I. below deep-root zone
            ET = 0.0
         ELSEIF (gw_depth .GT. depth_srz) THEN
c II. within deep-root zone
            ET = K*et_sat_max / 12. * KC * (depth_drz-gw_depth)/
     $      (depth_drz-depth_srz)
         ELSE
cIII. within shallow-root zone
            ET = K * et_sat_max / 12. * KC
         ENDIF
         et_sat(node) = ET
c correction for unaccounted for unsaturated et
cxcx         et_unsat_unacct(node) = et_unsat_adj(node) - et_unsat_actual(node)
         et_sat(node) = et_sat(node) + et_unsat_unacct(node)
C
      ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
C LAND IS FLOODED
         SWD = ponding
         IF (SWD .GT. OWPOND2(node)) THEN
c  V. above ground, above OWPOND
            ET = KMAX(lutyp(node)) * et_sat_max /12.
         ELSE
c IV. above ground, below OWPOND
            ET = (SWD/OWPOND2(node)*(KC * (KMAX(lutyp(node))
     $      -K)) + K * KC) * et_sat_max / 12.
         ENDIF
         et_ponding(node) = ET
c correction for unaccounted for unsaturated et
cxcx         et_unsat_unacct(node) = et_unsat_adj(node) - et_unsat_actual(node)
         et_ponding(node) = et_ponding(node) + et_unsat_unacct(node)
         IF (et_ponding(node) .GT. ponding)THEN
            et_sat(node) = et_ponding(node) - ponding
            et_ponding(node) = ponding
         ENDIF
      ENDIF
      RETURN
   20 FORMAT('Unaccounted for ETU at node:',i5, ' or (x,y):',2i3,' is '
     $,f8.4,/,6x, 'iyear:',i5,' month:',i3,' jday:',i10)
      END
```

lec_et_initial_read.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lec_et_initial_read.F,v 1.5 2003/07/03 20:07:31 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et_initial_read.F,v $
C   $Name: rel-5-5 $
C
C
C  Subroutine OPENs files used for the LEC ET routine
C  and initializes other grid_io CALLs.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_et_initial_read (ism, isyretbin, ibm, getfdd,
     & datadir)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trigger.inc'

      CHARACTER*200 datadir, tempfname

      INTEGER getfd, getfdd(150), ierr

      EXTERNAL getfd

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      READ(use_lec_et,*)
      READ(use_lec_et,*)
      READ(use_lec_et,*) num_use_type,etiu_unrest_PRINT_opt

      IF (num_use_type .GT. max_n_type) THEN
```

```
         WRITE(*,20) num_use_type, max_n_type
         STOP
      ENDIF
c Read the input file list
      READ(use_lec_et,*)
      READ(use_lec_et,*)
      READ(use_lec_et,*)
      READ(use_lec_et,10) et_in_unit, et_in_file
      READ(use_lec_et,10) net_irr_unit, net_irr_file
c Now READ the output file list
      READ(use_lec_et,*)
      READ(use_lec_et,*)
      READ(use_lec_et,*)
c
c      READ(use_lec_et,1) et_comp_unit, et_comp_file
      READ(use_lec_et,10) et_comp_unit, tempfname
      ibeg = iscan(tempfname,200,-200,'/',1,1,ipos2)
      et_comp_file = DATADIR(1:LENGTH(DATADIR)) // '/' // tempfname(ibeg
     $ + 1:LENGTH(tempfname))
c
c      READ(use_lec_et,1) supply_unit, supply_file
      READ(use_lec_et,10) supply_unit, tempfname
      ibeg = iscan(tempfname,200,-200,'/',1,1,ipos2)
      supply_file = DATADIR(1:LENGTH(DATADIR)) // '/' // tempfname(ibeg
     $ + 1:LENGTH(tempfname))
c
ccjn>>
      IF (use_trigger .NE. 0) THEN
c
c        READ(use_lec_et,1) shortage_unit, shortage_file
         READ(use_lec_et,10) shortage_unit, tempfname
         ibeg = iscan(tempfname,200,-200,'/',1,1,ipos2)
         shortage_file = DATADIR(1:LENGTH(DATADIR)) // '/' // tempfname
     $   (ibeg+1:LENGTH(tempfname))
c
c        READ(use_lec_et,1) est_et_unit, est_et_file
         READ(use_lec_et,10) est_et_unit, tempfname
         ibeg = iscan(tempfname,200,-200,'/',1,1,ipos2)
         est_et_file = DATADIR(1:LENGTH(DATADIR)) // '/' // tempfname
     $   (ibeg+1:LENGTH(tempfname))
c
         IF (etiu_unrest_PRINT_opt .EQ. 'YES') THEN
            READ(use_lec_et,10) est_unres_et_unit, tempfname
            ibeg = iscan(tempfname,200,-200,'/',1,1,ipos2)
            est_unres_et_file = DATADIR(1:LENGTH(DATADIR)) // '/' //
     $      tempfname(ibeg+1:LENGTH(tempfname))
         ENDIF
      ENDIF
      CLOSE(use_lec_et)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c Open and initialize the ET input grid file

c
c  first compute the number of binary file snapshots to skip for
c  positioning binary files to correct starting date.
c
      nskip = iymdjl(ifyr,ibm,1) - iymdjl(isyretbin,ism,1)
```

```
      CALL openfilef77(et_in_unit,et_in_file,"rb")
      getfdd(et_in_unit) = et_in_unit
      et_in_def = et_in_unit
      CALL gridrhd(et_in_def, ierr)
      CALL getgrid(title_et_in, nrows, num_lec_nodes, x_size, y_size,
     $min_x_lec, max_x_lec, isum_lec)
c
      IF (nskip.GT.0)THEN
         CALL gridskp(et_in_def, nskip*3, ierr)
      ENDIF
c

c Call map_lec_node_to_model to map the LEC developed area nodes into the
c model nodes.  This is done only after the first getgrid CALL for the LEC.

      CALL map_lec_node_to_model

cc Open and initialize the potential ET grid file
c Open and initialize the net irrigation demand grid file
      CALL openfilef77(net_irr_unit,net_irr_file,"rb")
      getfdd(net_irr_unit) = net_irr_unit
      net_irr_def = net_irr_unit
      CALL gridrhd(net_irr_def, ierr)
      CALL getgrid(title_net_irr, nrows, num_lec_nodes, x_size, y_size,
     $min_x_lec, max_x_lec, isum_lec)
c
      IF (nskip.GT.0)THEN
         CALL gridskp(net_irr_def, nskip*num_use_type, ierr)
      ENDIF
c

c Now OPEN and initialize the grid output file for
c unsaturated, unsat_irr, unsat_nonirr, ponding, and saturated ET

      CALL openfilef77(et_comp_unit,et_comp_file,"wb")
      getfdd(et_comp_unit) = et_comp_unit
      et_comp_def = et_comp_unit
      CALL setgrid('SFWMM Output ET Components Data (in)      ', maxy,
     $isum(maxy+1), dx, dy, minx, maxx, isum)
      CALL gridwhd(et_comp_def, ierr)

c Now OPEN and initialize the output file for supply

      CALL openfilef77(supply_unit,supply_file,"wb")
      getfdd(supply_unit) = supply_unit
      supply_def = supply_unit
      CALL setgrid('SFWMM Output Supply Data (pws:mgd, irr:in)', maxy,
     $isum(maxy+1), dx, dy, minx, maxx, isum)
      CALL gridwhd(supply_def, ierr)

c If trigger module is active, Open and initialize output file
c for shortages
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (use_trigger .NE. 0) THEN
         CALL openfilef77(shortage_unit,shortage_file,"wb")
         getfdd(shortage_unit) = shortage_unit
         shortage_def = shortage_unit
```

```
         CALL setgrid('SFWMM Output Shortage Data (in)         ', maxy,
     $   isum(maxy+1), dx, dy, minx, maxx, isum)
         CALL gridwhd(shortage_def, ierr)
      ENDIF

c If trigger module is active, Open and initialize output file
c for estimated et (used for LEC DA post processing)

      IF (use_trigger .NE. 0) THEN
         CALL openfilef77(est_et_unit,est_et_file,"wb")
         getfdd(est_et_unit) = est_et_unit
         est_et_def = est_et_unit
         CALL setgrid('SFWMM Output Estimated ET Data (in)     ', maxy,
     $   isum(maxy+1), dx, dy, minx, maxx, isum)
         CALL gridwhd(est_et_def, ierr)
      ENDIF
      IF (use_trigger .NE. 0 .AND. etiu_unrest_PRINT_opt .EQ. 'YES')
     $THEN
         CALL openfilef77(est_unres_et_unit,est_unres_et_file,"wb")
         getfdd(est_unres_et_unit) = est_unres_et_unit
         est_unres_et_def = est_unres_et_unit
         CALL setgrid('SFWMM Output Estimated Unrestricted Data (in)   '
     $   , maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
         CALL gridwhd(est_unres_et_def, ierr)
      ENDIF
      RETURN
   10 FORMAT(i4,1x,a120)
   20 FORMAT('number of types =',i3,' > DIMENSION',i3)
      END
```

lec_et_set_cutbacks.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lec_et_set_cutbacks.F,v 1.4 2003/07/03 20:07:31 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et_set_cutbacks.F,v $
C   $Name: rel-5-5 $
C
C
C  When LEC ET is used, SUBROUTINE LEC_ET_SET_CUTBACKS is CALLed each day for
C  each cell in the LEC developed area.  The SUBROUTINE computes values for
C  daily net irrigation supply (per irrigation use type), daily shortages per
C  irrigation use type,
C  daily total net irrigation demand, daily total net irrigation supply, and
C  daily total shortages.  Daily adjusted et in the unsaturated zone for
C  irrigated ares, et_unsat_irr_adj, also computed.
C  (the daily values are summed monthly later in SUBROUTINE
lec_et_sum_monthlies)
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_et_set_cutbacks (lec_node)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'

      INTEGER lec_node, model_node, type

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      model_node = model_node_in_lec(lec_node)
      L = model_node

C  be sure to initialize all necessary variables for totals
```

```
      net_irr_t(L) = 0.
      supply_t(L) = 0.
      shortage_t(L) = 0.

      DO type = 1, num_use_type

c compute daily irrigation supply using same daily distribution
c as the net_irrigation demand
clmb for testing model w/o trigger module, use next line
clmb        supply(L,type) = net_irr(L,type)

         IF ( net_irr_month(L,type) .EQ. 0.0 ) THEN
            supply(L,type) = 0.0
         ELSE
            supply(L,type) = supply_month(L,type) *
     $      net_irr(L,type) / net_irr_month(L,type)
         ENDIF
         shortage(L,type) = net_irr(L,type) - supply(L,type)
c
c accumulate totals for all irrigation use types for the cell
         net_irr_t(L) = net_irr_t(L) +  net_irr(L,type)
         supply_t(L) = supply_t(L) + supply(L,type)
         shortage_t(L) = shortage_t(L) + shortage(L,type)
      END DO
c
c compute daily adjusted unsaturated zone et for irrigated areas
c using the same daily distribution as the etiu that is input
clmb for testing model w/o trigger module, use next line
clmb      frac_etiu_red = 0.0
      IF ( et_unsat_irr_month(L) .EQ. 0.0 ) THEN
         frac_etiu_red = 1.0
      ELSE
         IF ( etiu_reduction_t(L) .GT. et_unsat_irr_month(L) ) THEN
clb etiu reduction due to regrn eqns are higher than total etiu !!
clb assume etiu goes down to zero, although rest of reduction can be
clb accounted for by etnu and/or ets
c        PRINT *,l,'etiu_reduction_t() exceeds et_unsat_irr_month'
c
            frac_etiu_red = 0.0
         ELSE
            frac_etiu_red = (1. - etiu_reduction_t(L)/
     $      et_unsat_irr_month(L))
         ENDIF
      ENDIF
      et_unsat_irr_adj(L) = frac_etiu_red * et_unsat_irr(L)
c assumption: reduction in etu is due only to reduction in etiu
      et_unsat_adj(L) = et_unsat(L) - (et_unsat_irr(L) -
     $et_unsat_irr_adj(L))
      RETURN
      END
```

lec_et_sum_monthlies.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lec_et_sum_monthlies.F,v 1.7 2003/07/03 20:07:32 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et_sum_monthlies.F,v $
C   $Name: rel-5-5 $
C
C
C  Subroutine LEC_ET_SUM_MONTHLIES sums monthly values
C
C  and WRITEs the following values to 4 composite binary files:
C  1.et_components.bin
C      et_unsat_month
C      et_unsat_irr_month
C      et_unsat_nonirr_month
C      et_ponding_month
C      et_sat_month
C
C  2.nirrsup.bin
C      supply_month_pws
C      supply_month (for irrigation use types 1)
C      supply_month (for irrigation use types 2)
C      supply_month (for irrigation use types 3)
C      supply_month (for irrigation use types 4)
C      supply_month (for irrigation use types 5)
C      supply_month (for irrigation use types 6)
C
C   The following composite binaries contain monthly snapshots
C   ONLY for months when water restrictions were in effect:
C
C  3.shortage.bin
C      shortage_month_pws
C      shortage_month (for all irrigation use types (6))
C
C  4.est_et.bin
C      est_et_unsat_irr         for irrigation use type 1
C      est_et_unsat_irr_restr   for irrigation use type 1
```

```
C      est_et_unsat_irr         for irrigation use type 2
C      est_et_unsat_irr_restr   for irrigation use type 2
C       .
C       .
C       .
C      est_et_unsat_irr         for last irrigation use type
C      est_et_unsat_irr_restr   for last irrigation use type
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_et_sum_monthlies (iyr, imth, max_day_in_month,
     & getfdd, rmonthly_loss_bmp,total_small_afact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'daydump.inc'

      CHARACTER*256 buffer

      INTEGER getfdd(150), iday, ierr, iyr, max_day_in_month, month,
     & node, type

      REAL values(max_ncells),total_small_afact(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      CALL dstring(month,iday,iyear+ifyr-1,buffer)

C If we're in the first day of the month, initialize the monthly sums.
C Note that the monthlies, some of which are written out, are converted
C back to inches.

      IF (iday .EQ. 1) THEN
         DO node = 1, max_ncells
ccjn>>
c make sure to only sum those values which need to be summed.  That is,
c some of the monthly sums are done outside this SUBROUTINE since they
c are needed for reasons other than output.
ccjn<<
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            infilt_month(node) = infilt(node) * 12.
            perc_month(node) = perc(node) * 12.
            et_poten_month(node) = et_poten(node) * 12.
     $      * (1.0 - total_small_afact(node) - carfact(node))
            et_ponding_month(node) = et_ponding(node) * 12.
            et_unsat_month(node) = et_unsat_actual(node) * 12.
     $      * (1.0 - carfact(node) - total_small_afact(node))
            et_unsat_unacct_month(node) = et_unsat_unacct(node) * 12.
```

```
     $      * (1.0 - carfact(node) - total_small_afact(node))
c
            et_unsat_irr_actual_month(node) = et_unsat_irr_actual(node)
     $      * 12. * (1.0 - carfact(node) - total_small_afact(node))
            et_unsat_nonirr_month(node) = et_unsat_nonirr_actual(node) *
     $      12. * (1.0 - carfact(node) - total_small_afact(node))
            et_sat_month(node) = et_sat(node) * 12.
            net_irr_t_month(node) = net_irr_t(node) * 12.
     $      * (1.0 - carfact(node) - total_small_afact(node))
            supply_t_month(node) = supply_t(node) * 12.
     $      * (1.0 - carfact(node) - total_small_afact(node))
            shortage_t_month(node) = shortage_t(node) * 12.
     $      * (1.0 - carfact(node) - total_small_afact(node))
            DO type = 1, num_use_type
               etiu_unrestr(node,type) = etiu_unrestr(node,type) * 12.
     $          * (1.0 - carfact(node) - total_small_afact(node))
               etiu_restr(node,type) = etiu_restr(node,type) * 12.
     $          * (1.0 - carfact(node) - total_small_afact(node))
            END DO
c
         END DO
c WRITE initial snapshot to grid files IF today is the first day of simulation
         IF (IMTH .EQ. IBM) THEN
            IF (.NOT. lp(19)) THEN
               CALL dstring(imth,iday,iyear+ifyr-1,buffer)
               CALL setgrid('SFWMM Unaccounted for Unsaturated ET Data '
     $         , maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
               CALL gwrite(57, buffer, et_unsat_unacct_month, ierr)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
               CALL setgrid('SFWMM Output ET Components Data (in)      '
     $         , maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
               CALL gwrite(et_comp_def, buffer, et_unsat_month, ierr)

               CALL gwrite(et_comp_def, buffer,
     $         et_unsat_irr_actual_month, ierr)
               CALL gwrite(et_comp_def, buffer, et_unsat_nonirr_month,
     $         ierr)
               CALL gwrite(et_comp_def, buffer, et_ponding_month, ierr)

               CALL gwrite(et_comp_def, buffer, et_sat_month, ierr)

c
               CALL setgrid('SFWMM Output Supply Data (pws:mgd, irr:in)'
     $         , maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
c
clb>> ff gwrite has to be activated once budget program is READy for it
c supply_month_pws will be output even IF trigger module is off
               CALL gwrite(supply_def, buffer, supply_month_pws, ierr)

               DO type = 1, num_use_type
                  CALL gwrite(supply_def, buffer, supply_month(1,type),
     $            ierr)
               END DO
            ENDIF
         ENDIF
      ELSE
```

```
         DO node = 1, max_ncells
            infilt_month(node) = infilt_month(node) + infilt(node)*12.

            perc_month(node) = perc_month(node) + perc(node)*12.

            et_poten_month(node) = et_poten_month(node) + et_poten(node)
     $      * 12. * (1.0 - carfact(node) - total_small_afact(node))
            et_ponding_month(node) = et_ponding_month(node) + et_ponding
     $      (node) * 12.
            et_unsat_month(node) = et_unsat_month(node) +
     $      et_unsat_actual(node) * 12.* (1.0 - carfact(node)
     $      - total_small_afact(node))
            et_unsat_unacct_month(node) = et_unsat_unacct_month(node) +
     $      et_unsat_unacct(node) * 12.* (1.0 - carfact(node)
     $      - total_small_afact(node))
c
            et_unsat_irr_actual_month(node) = et_unsat_irr_actual_month
     $      (node) + et_unsat_irr_actual(node) * 12.
     $      * (1.0 - carfact(node) - total_small_afact(node))

            et_unsat_nonirr_month(node) = et_unsat_nonirr_month(node) +
     $      et_unsat_nonirr_actual(node) * 12.* (1.0 - carfact(node)
     $      - total_small_afact(node))
            et_sat_month(node) = et_sat_month(node) + et_sat(node)*12.

            net_irr_t_month(node) = net_irr_t_month(node) + net_irr_t
     $      (node) * 12. * (1.0-carfact(node)-total_small_afact(node))
            supply_t_month(node) = supply_t_month(node) + supply_t(node)
     $      * 12. * (1.0 - carfact(node) - total_small_afact(node))
            shortage_t_month(node) = shortage_t_month(node) + shortage_t
     $      (node)* 12. * (1.0-carfact(node)-total_small_afact(node))
         END DO
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c If we're in the last day of the month, WRITE out the monthly values.
c Do this for ET (unsat, unsat_irr, unsat_nonirr, ponding & sat)
c and supply by (use type)

      IF (iday .EQ. max_day_in_month) THEN
         IF (.NOT. lp(19)) THEN
            DO node = 1,max_ncells
               IF (ibsn(node) .GE. 7 .AND. ibsn(node) .LE. 9 .AND.
     $         land_use_type(LUTYP(node)) .NE. 'WETLAND') THEN

                  et_ponding_month(node) = et_ponding_month(node) +
     $            (rmonthly_loss_bmp * 43560. / ((nagnode(1) +
     $            nagnode(2) + nagnode(3)) * gdar) ) * 12.
                  etm(node) = etm(node) + rmonthly_loss_bmp * 43560.
     $            /((nagnode(1) + nagnode(2) + nagnode(3)) * gdar)
               ENDIF
            END DO
            CALL setgrid(
     $      'SFWMM Unaccounted for Unsaturated ET Data (in)', maxy, isum
     $      (maxy+1), dx, dy, minx, maxx, isum)
            CALL gwrite(57, buffer, et_unsat_unacct_month, ierr)
c
            CALL setgrid(
```

```
     $      'SFWMM ET Components Output Data (in)          ', maxy, isum
     $      (maxy+1), dx, dy, minx, maxx, isum)
            CALL gwrite(et_comp_def, buffer, et_unsat_month, ierr)
c
            CALL gwrite(et_comp_def, buffer, et_unsat_irr_actual_month,
     $      ierr)
            CALL gwrite(et_comp_def, buffer, et_unsat_nonirr_month,
     $      ierr)
            CALL gwrite(et_comp_def, buffer, et_ponding_month, ierr)
c
            CALL gwrite(et_comp_def, buffer, et_sat_month, ierr)
c
            CALL setgrid('SFWMM Output Supply Data (in)             ',
     $      maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
c
clb>> ff gwrite has to be activated once budget program is READy for it
c supply_month_pws will be output even IF trigger module is off
            CALL gwrite(supply_def, buffer, supply_month_pws, ierr)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c
            DO type = 1, num_use_type
               DO iii = 1,max_ncells
                  supply_month(iii,type) = supply_month(iii,type) * 12.
               END DO
               CALL gwrite(supply_def, buffer, supply_month(1,type),
     $         ierr)
            END DO
            IF (etiu_unrest_PRINT_opt .EQ. 'YES') THEN
               CALL setgrid(
     $         'SFWMM Output Estimated Unrestricted Data (in) ', maxy,
     $         isum(maxy + 1), dx, dy, minx, maxx, isum)
               DO type = 1, num_use_type
                  CALL gwrite(est_unres_et_def, buffer, etiu_unrestr
     $            (1,type), ierr)
               END DO
            ENDIF
         ENDIF
c IF trigger module is being used AND IF lec water restrictions were
c in effect for the month, THEN WRITE the monthly values of irrigation
c shortages and estimated unsat zone et for all the irrigation use types.

         IF ( use_trigger .GT. 0 .AND. lec_restrictions1 ) THEN
            CALL setgrid('SFWMM Output Shortage Data (in)         ',
     $      maxy, isum(maxy + 1), dx, dy, minx, maxx, isum)
c
clb>> ff gwrite has to be activated once budget program is READy for it
c shortage_month_pws will not be output IF trigger module is off
c IF desired, it has to be written outside this if-ENDIF block
            CALL gwrite(shortage_def, buffer, shortage_month_pws, ierr)

clb<<
            DO type = 1, num_use_type
               DO iii = 1,max_ncells
                  shortage_month(iii,type) = shortage_month(iii,type)
     $            * 12.
               END DO
               CALL gwrite(shortage_def, buffer, shortage_month(1,type),
```

```
     $         ierr)
            END DO

            CALL setgrid('SFWMM Output Estimated ET Data (in)     ',
     $      maxy, isum(maxy + 1), dx, dy, minx, maxx, isum)
            DO type = 1, num_use_type
               CALL gwrite(est_et_def, buffer, etiu_unrestr(1,type),
     $         ierr)
               CALL gwrite(est_et_def, buffer, etiu_restr(1,type), ierr)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            END DO
            lec_restrictions1 = .FALSE.
         ENDIF
      ENDIF
      IF ( lp(19) .AND.  iddump ) THEN
         CALL setgrid('SFWMM Unaccounted for Unsaturated ET Data (in)',
     $   maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
         DO iii = 1,max_ncells
            values(iii) = et_unsat_unacct(iii) * 12.
         END DO
         CALL gwrite(57, buffer, values, ierr)
c
         CALL setgrid('SFWMM ET Components Output Data (in)          ',
     $   maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
         DO iii = 1,max_ncells
            values(iii) = et_unsat_actual(iii) * 12.
         END DO
         CALL gwrite(et_comp_def, buffer, values, ierr)
         DO iii = 1,max_ncells
            values(iii) = et_unsat_irr_actual(iii) * 12.
         END DO
         CALL gwrite(et_comp_def, buffer, values, ierr)
         DO iii = 1,max_ncells
            values(iii) = et_unsat_nonirr_actual(iii) * 12.
         END DO
         CALL gwrite(et_comp_def, buffer, values, ierr)
         DO iii = 1,max_ncells
            values(iii) = et_ponding(iii) * 12.
         END DO
         CALL gwrite(et_comp_def, buffer, values, ierr)
         DO iii = 1,max_ncells
            values(iii) = et_sat(iii) * 12.
         END DO
         CALL gwrite(et_comp_def, buffer, values, ierr)
c
         CALL setgrid('SFWMM Output Supply Data (in)             ',
     $   maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c supply_month_pws will be output even IF trigger module is off
         CALL gwrite(supply_def, buffer, supply_month_pws, ierr)
c
         DO type = 1, num_use_type
            DO iii = 1,max_ncells
               values(iii) = supply(iii,type) * 12.
            END DO
            CALL gwrite(supply_def, buffer, values, ierr)
```

```
   END DO
ENDIF

RETURN
END
```

lec_et_time_series_read.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lec_et_time_series_read.F,v 1.4 2003/07/03 20:07:32 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_et_time_series_read.F,v $
C   $Name: rel-5-5 $
C
C
C  Subroutine READs daily values for the LEC ET input grid files
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_et_time_series_read (iday)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'

      EXTERNAL getfd

      INTEGER getfd, iday, ierr, type

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C Read input ET data: PET, unsaturated zone ET for the entire cell, and
C  unsaturated zone ET for irrigated areas within each cell

C READ lec potential evapotranspiration
      CALL setgrid(title_et_in, nrows, num_lec_nodes, x_size, y_size,
     $ min_x_lec, max_x_lec, isum_lec)
      CALL gread(et_in_def, snap_tag, et_poten_lec, ierr)
c map to model array and convert inches to feet
      CALL map_lec_array_to_model(et_poten_lec, et_poten)
c
c READ lec unsaturated zone evapotranspiration,
```

```
c both irrigated and non-irrigated areas
      CALL gread(et_in_def, snap_tag, et_unsat_lec, ierr)
c map to model array and convert inches to feet
      CALL map_lec_array_to_model(et_unsat_lec, et_unsat)
c
c READ lec unsaturated zone evapotranspiration, irrigated areas only
      CALL gread(et_in_def, snap_tag, et_unsat_lec, ierr)
c map to model array and convert inches to feet
      CALL map_lec_array_to_model(et_unsat_lec, et_unsat_irr)
c
c calculate unsaturated zone evapotranspiration, non-irrigated areas only
      DO i = 1,max_ncells
         et_unsat_nonirr(i) = et_unsat(i) - et_unsat_irr(i)
         IF (et_unsat_nonirr(i) .LT. -0.0002) THEN
            WRITE(*,10)i,iday
            STOP
         ENDIF
         IF (et_unsat_nonirr(i) .LT. 0.) THEN
            et_unsat_irr(i) =  et_unsat(i)
            et_unsat_nonirr(i) = 0.0
         ENDIF
      END DO
c
c READ net irrigation demands for all use types
      CALL setgrid(title_net_irr, nrows, num_lec_nodes, x_size, y_size,
     $ min_x_lec, max_x_lec, isum_lec)
      DO type = 1, num_use_type
         CALL gread(net_irr_def, snap_tag, net_irr_lec, ierr)
cbw gread above ??
c map to model array and convert inches to feet
         CALL map_lec_array_to_model(net_irr_lec, net_irr(1,type))
      END DO
c
      RETURN
   10 FORMAT(1x,'etiu is greater than etu in node:',i5, ',day: ',i5)
      END
```

lec_pws_irrig.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lec_pws_irrig.F,v 1.10 2005/06/10 13:29:11 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lec_pws_irrig.F,v $
C   $Name: rel-5-5 $
C
C-----------------------------------------------------------------------------
C  This subroutine computes public water supplies and shortages
C  as well as net irrigation and shortages in developed areas
C  if trigger module is used and water restrictions are in effect
C  for current month
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lec_pws_irrig (mon_pet0, month, iyr, wfld,
     $pump_def_semhlywd, pump_semhlywd)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'stat.inc'

      REAL mon_pet0(max_ncells)
      CHARACTER wfld(max_n_wells)*9

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C Call subr. reduce_wellq to reduce public water
C supply pumpage IF trigger module is to be used AND IF lec
C water restrictions are in effect for the current month.

      IF (use_trigger .GE. 1 .AND. lec_restrictions) THEN

C set lec_restrictions1 = .TRUE. so that gwrite to shortage and est_et
C binary files (see subroutine lec_et_sum_monthlies) synchronize with
```

```
C trigger output file trigwell (see subroutine reduce_wellq)

         lec_restrictions1 = .TRUE.

C compute public water supply supplies and shortages
C (store original pump() into pump_def and reset pump() = to supply)
         pump_def_semhlywd = 0.0
         pump_semhlywd = 0.0
         DO iwell = 1,nwell
            pump_def(iwell,month) = pump(iwell,month)
            IF (WFLD(iwell) .eq. '06-00001-')  pump_def_semhlywd =
     $         pump_def_semhlywd +  pump_def(iwell,month)
           call reduce_wellq(iyr, month, locw(iwell), 0, iwell)
C note: well no. iwell is needed only for pws wells
            pump(iwell,month) = pump_def(iwell,month) * fract_wellq(locw
     $      (iwell))
            IF (WFLD(iwell) .eq. '06-00001-')  pump_semhlywd =
     $         pump_semhlywd +  pump(iwell,month)
         END DO
         DO iwell = 1,nwell
            supply_month_pws(locw(iwell)) = supply_month_pws(locw(iwell)
     $      ) + pump(iwell,month)
            shortage_month_pws(locw(iwell)) = shortage_month_pws(locw
     $      (iwell)) + pump_def(iwell,month) - pump(iwell,month)
         END DO
c IF we are using lec_et data, compute irrigation supplies & shortages
         IF (use_lec_et .NE. 0) THEN
            DO lec_node = 1,num_lec_nodes
               model_node = model_node_in_lec(lec_node)
               L = model_node
               IF ( ibsn(L) .NE. 40 ) THEN
                  DO itype = 1,num_use_type
                     IF (irrig_area(L,itype) .GT. 0.) THEN
                        call reduce_wellq(iyr, month, L, itype, 0)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c note: the zero (0) at the END of the argument list is a dummy variable
c       for irrigation supplies & shortages
                        supply_month(L,itype) = amin1(net_irr_month
     $                  (L,itype),nirrmax(L,itype))
                        shortage_month(L,itype) = net_irr_month(L,itype)
     $                  - supply_month(L,itype)
c IF there is a shortage for the current irrigation use type, THEN
c compute the estimated reduction in unsaturated zone et using regression
c equations developed to mimick AFSIRS.
                        IF (shortage_month(L,itype) .GT. 0.000001) THEN
                           cellirr = net_irr_month(L,itype) * 12. /
     $                     frac_irr_area(L,itype)
                           etiu_unrestr(L,itype) = est_etiu( itype,
     $                     cellirr, RFM(L),mon_pet0(ietzon(L))) / 12. *
     $                     frac_irr_area(L,itype)

                           cellirr = supply_month(L,itype) * 12. /
     $                     frac_irr_area(L,itype)
                           etiu_unrestr(L,itype) = est_etiu(itype,
     $                     cellirr, RFM(L),mon_pet0(ietzon(L))) / 12. *
     $                     frac_irr_area(L,itype)
```

```
                            cellirr = supply_month(L,itype)*12. /
     $                      frac_irr_area(L,itype)
                            etiu_restr(L,itype) = est_etiu( itype,
     $                      cellirr, RFM(L),mon_pet0(ietzon(L))) / 12. *
     $                      frac_irr_area(L,itype)

                            etiu_reduction(L,itype) = etiu_unrestr
     $                      (L,itype) - etiu_restr(L,itype)
                            etiu_reduction_t(L) = etiu_reduction_t(L) +
     $                      etiu_reduction(L,itype)*(1.0-carfact(L))
                         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c case wherein there should be no irrig demand (& so is supply) for cells
c which have no irrigated areas for use type "itype"
c no need to zero supply_month(L,itype) and shortage(L,itype) here since
c they are already initialized upon entry to month loop
c this part of the if-ELSE-ENDIF construct is used just to verify that
c there shouldn't be any irrig demands for a use type that has zero acreage
c in a given cell
                       ENDIF ! loop for irrig_area(L,itype) > 0 test
                    END DO   !END loop for irrigation use types
                 ENDIF       !ENDIF for subset of lec_nodes
              END DO         !END loop for lec_nodes
           ENDIF             !ENDIF for use_lec_et
        ENDIF                !ENDIF for use_trigger AND .lec_restrictions.
c
c  update well pumpage array
c
        DO 10 IW = 1, NWELL
           WELLS(LOCW(IW)) = PUMP(IW,MONTH) + WELLS(LOCW(IW))
   10 CONTINUE
        DO    iw = 1,nwell_ind_rss
           wells(loc_ind_rss(iw)) = pump_ind_rss(iw,month) + wells
      $   (loc_ind_rss(iw))
        END DO
        RETURN
        END
```

lineout.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lineout.F,v 1.6 2003/07/03 20:07:33 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lineout.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lineout

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'

      DIMENSION ixpt(35), iypt(35)

      DATA ixpt /1, 2, 3, 4, 5, 6, 7, 8, 9,10,11,12,13,14,15,16, 17,18,
     & 19,20,21,22,23,24,25,26,27,28,29,30,31,32, 33,34,35/
      DATA iypt /27,26,26,26,26,26,25,25,25,25,25,25,25,24,23,22, 22,22,
     & 22,22,22,22,22,22,22,22,22,22,22,22,22,22, 23,23,23/
      DATA npts /35/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      NODE(JX,JY) = JX - MINX(JY) + 1 + ISUM(JY)
      WRITE(56,10) MONTH,IDAY,IYEAR+IFYR-1,(H(NODE(IXPT(I),IYPT (I)))
     $+POND(NODE(IXPT(I),IYPT(I))),I=1,NPTS)
      RETURN
   10 FORMAT(2(I2,'/'),I4,1X,35F5.2)
      END
```

locate.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords:
C   $Author: rnovoa $
C   $Id: locate.F,v 1.3 2003/07/03 20:07:33 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/locate.F,v $
C   $Name: rel-5-5 $
C
C
C  given an array xx of length n, and given a value of x, RETURNs a value
C  of j such that x is between xx(j) and xx(j+1). xx must be monotonic,
C  either increasing or decreasing. j = 0 or j = n is RETURNed
C  to indicate that x is out of range
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE locate (xx, n, x, j)

      DIMENSION xx(n)

      jl = 0
      ju = n+1
   10 IF (ju-jl .GT. 1)THEN
         jm = (ju+jl)/2
         IF ((xx(n) .GT. xx(1)) .EQV. (x .GT. xx(jm)))THEN
            jl = jm
         ELSE
            ju = jm
         ENDIF
         goto 10
      ENDIF
      j = jl

      RETURN
      END
```

locwslwdd.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: locwslwdd.F,v 1.12 2005/06/10 13:29:11 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/locwslwdd.F,v $
C   $Name: rel-5-5 $
C
C
C  DETERMINE WATER DELIVERIES INTO LAKE WORTH DRAINAGE DISTRICT (LWDD) FROM
C  LOCAL SOURCES SUCH AS C-51 BASIN AND EXCESS WATER FROM LWDD CANALS.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE locwslwdd (wsnde1, wsnde2, wsnde3, wsndes, wsndese,
     & crmin, volwtr, prevqu, rain, istr_count, rlocal_ws_to_e1,
     & rlocal_ws_to_e2, rlocal_ws_to_e3, rlocal_ws_to_es,
     & rlocal_ws_to_ese)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'STRCOUNT.inc'

      DIMENSION crmin(max_ncnls,max_n_seas), prevqu(max_ncnls),
     & rain(max_ncells), volwtr(max_n_res)

      DOUBLE PRECISION crmin,stg_lim_c51_excws,stg_lim_hlsb_excws
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
      WSNDE1 = DQU(IE1) - (DQU(IE2) + DQU(IE3))
      WSNDE2 = DQU(IE2) - DQU(IESE)
      WSNDE3 = DQU(IE3)
      WSNDES = DQU(IES)
      WSNDESE= DQU(IESE)
      istr_count = nflpts

C FIRST MEET THE WATER SUPPLY NEEDS FROM WITHIN THE
C LWDD WITH EXCESS WATER,IF ANY

      IF (WSNDE1 .LT. 0.0) THEN
         TDOWNWS = DQU(IE2) + DQU(IE3)
         QU(IE1) = QU(IE1) - AMIN1(ABS(WSNDE1),TDOWNWS) * 86400.

         QU(IE3) = QU(IE3) + AMIN1(DQU(IE3),ABS(WSNDE1)) * 86400

         WSNDE3  = WSNDE3 - AMIN1(DQU(IE3),ABS(WSNDE1))
         WSAVE2  = ABS(WSNDE1) - AMIN1(DQU(IE3),ABS(WSNDE1))
         QU(IE2) = QU(IE2) + AMIN1(WSAVE2,DQU(IE2)) * 86400.
         WSNDE2  = WSNDE2  - AMIN1(WSAVE2,DQU(IE2))
         WSNDE1  = WSNDE1  + AMIN1(ABS(WSNDE1),TDOWNWS)
      ENDIF
C
      IF (WSNDE2 .LT. 0.0) THEN
         QU(IE2) = QU(IE2) - AMIN1(ABS(WSNDE2),DQU(IESE)) * 86400.

         QU(IESE) = QU(IESE) + AMIN1(ABS(WSNDE2),DQU(IESE)) * 86400.

         WSNDESE  = WSNDESE - AMIN1(ABS(WSNDE2),DQU(IESE))
         WSNDE2  = WSNDE2  + AMIN1(ABS(WSNDE2),DQU(IESE))
      ENDIF
C
      IF (WSNDES .LT. 0.0) THEN
         QU(IES) = QU(IES) - ABS(WSNDES) * 86400.
         QU(IESE) = QU(IESE) + ABS(WSNDES) * 86400.
         WSNDESE  = WSNDESE  + WSNDES
         WSNDES  = 0.0
      ENDIF
C
C  FROM C-51 AND E-4
C
C  AVAILABLE VOLUME OF WATER IN C-51 AND E-4 FOR SUPPLY
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      IF (runmode .EQ. 'CALIB') THEN
         HISTOUTF = KFLO(IFLS155)
         DO IS = 1,NSTRCTR(IC51)
            IF (TMPNAME(IC51,IS) .EQ. 'S155  ') THEN
               HISTOUTF = 0.0
               GO TO 10
            ENDIF
         ENDDO
      ELSE
         HISTOUTF = 0.0
      ENDIF
C
```

```
c10   IF (DQU(IC51) .EQ. 0.0) THEN
c      FLC51W = STROUT(CHDEP(IC51W),IC51W,IC51W,1,0,CRMIN
c    $            ,VOLWTR,PREVQU,RAIN,0)*86400. + DRAINC51
c     ELSE
c      FLC51W = DRAINC51
c     ENDIF
   10 stg_lim_c51_excws = max(RMNSTGS(IC51,1,iseas),CRMIN(ic51,iseas))
      AVVOLWS = (CHDEP(IC51) - stg_lim_c51_excws) * CL(IC51) * WIDTH
     $(IC51) + DRAINC51 + CGSEEP(IC51,stg_lim_c51_excws,0,0.)+TOVFLC
     $(IC51,stg_lim_c51_excws,0,0.,0,1) - HISTOUTF * 86400.
      AVVOLWS = AMAX1(AVVOLWS,0.0) / 86400.
C
C  TOTAL NEEDS TO BE MET BY C-51 AND E-4
C
      TNEEDC51 = WSNDE1 + WSNDE2 + WSNDE3
C
      IF (TNEEDC51 .GT. 0.0) THEN
         RATIOC51 = AMIN1(AVVOLWS/TNEEDC51,1.0)
      ELSE
         RATIOC51 = 1.0
      ENDIF
C
C  FROM C.S. 4
C
      RMAXWSPL = WSNDE3 * RATIOC51
      KFLO(IFLCS4) = AMIN1(CAPACCS4,RMAXWSPL)
      IF (RMAXWSPL .GT. 0.0) THEN
         RATIOWS = AMIN1(CAPACCS4/RMAXWSPL,1.0)
      ELSE
         RATIOWS = 1.0
      ENDIF
      QU(IE3) = QU(IE3) + WSNDE3 * RATIOC51 * RATIOWS * 86400.
      rlocal_ws_to_e3 = RATIOC51 * RATIOWS  *WSNDE3  * 86400.
      WSNDE3 = (1 - RATIOWS*RATIOC51) * WSNDE3
C
C  FROM C.S. 2
C
      RMAXWSPL = (WSNDE3 + WSNDE1) * RATIOC51
      IF (RMAXWSPL .GT. 0.0) THEN
         RATIOWS = AMIN1(CAPACCS2/RMAXWSPL,1.0)
      ELSE
         RATIOWS = 1.0
      ENDIF
      KFLO(IFLCS2) = AMIN1(CAPACCS2,RMAXWSPL)
      IF (runmode .EQ. 'CALIB') THEN
         istr_count = istr_count + 1
         istr_index(istr_count) = IFLCS2
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLCS2)
      ENDIF
C
C        JSTR = JSTR + 1
C        STRFLW(JSTR,IDSS) = KFLO(IFLCS2)
C
      QU(IE1) = QU(IE1) - WSNDE3 * RATIOC51 * RATIOWS * 86400.
      QU(IE3) = QU(IE3) + WSNDE3 * RATIOC51 * RATIOWS * 86400.
      rlocal_ws_to_e1 = RATIOWS * RATIOC51 * WSNDE1 * 86400.
```

```
      rlocal_ws_to_e3 = RATIOWS * RATIOC51 * WSNDE3 * 86400.
      WSNDE1 = (1 - RATIOWS * RATIOC51) * WSNDE1
      WSNDE3 = (1 - RATIOWS*RATIOC51) * WSNDE3
C
C  FROM C.S. 12
C
      WSNDE2WS = WSNDE2 * RATIOC51
      KFLO(IFLCS12) = AMIN1(CAPACCS12,WSNDE2WS)
      WSNDE2WS = WSNDE2WS - KFLO(IFLCS12)
      IF (runmode .EQ. 'CALIB') THEN
         istr_count = istr_count + 1
         istr_index(istr_count) = IFLCS12
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLCS12)
      ENDIF
C
C      JSTR = JSTR + 1
C      STRFLW(JSTR,IDSS) = KFLO(IFLCS12)
C
C  FROM C.S. 9
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      KFLO(IFLCS9) = AMIN1(CAPACCS9,WSNDE2WS)
      WSNDE2 = WSNDE2 - KFLO(IFLCS12) - KFLO(IFLCS9)
      rlocal_ws_to_e2 = (KFLO(IFLCS12) + KFLO(IFLCS9)) * 86400.
      IF (runmode .EQ. 'CALIB') THEN
         istr_count = istr_count + 1
         istr_index(istr_count) = IFLCS9
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLCS9)
      ENDIF
C
C      JSTR = JSTR + 1
C      STRFLW(JSTR,IDSS) = KFLO(IFLCS9)
C
C  FROM HILLSBORO CANAL
C
C  AVAILABLE VOLUME IN HILLSBORO CANAL FOR WATER SUPPLY
C
      stg_lim_hlsb_excws = max(RMNSTGS(LHL,1,iseas),CRMIN(LHL,iseas))
      AVVOLHL = (CHDEP(LHL) - stg_lim_hlsb_excws)*CL(LHL)*WIDTH(LHL)+
     $REGS39 * 86400. + CGSEEP(LHL,stg_lim_hlsb_excws,0,0.)+TOVFLC(LHL
     $,stg_lim_hlsb_excws,0,0.,0,1) + DRAINHIL
      AVVOLHL = AMAX1(AVVOLHL,0.0) / 86400.
C
C  DETERMINE NEED TO BE MET BY HILLSBORO CANAL
C
      TNEEDCS17W = WSNDE1
      TNEEDHILL = TNEEDCS17W + WSNDES + WSNDESE
      IF (TNEEDHILL .GT. 0.0) THEN
         RATIOHL = AMIN1(AVVOLHL/TNEEDHILL,1.0)
      ELSE
         RATIOHL = 1.0
      ENDIF
C
      WSDELCS17W = TNEEDCS17W * RATIOHL
      WSDELCS17W = AMIN1(WSDELCS17W,CAPACCS17W)
```

```
      rlocal_ws_to_e1 = rlocal_ws_to_e1 + WSDELCS17W * 86400.
      WSNDE1 = WSNDE1 - WSDELCS17W
      KFLO(IFLCS17W) = WSDELCS17W
      IF (runmode .EQ. 'CALIB') THEN
         istr_count = istr_count + 1
         istr_index(istr_count) = IFLCS17W
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLCS17W)
      ENDIF
C
C      JSTR = JSTR + 1
C      STRFLW(JSTR,IDSS) = KFLO(IFLCS17W)
C
      TNEEDCS17E = WSNDES + WSNDESE
      WSALLCS17E = TNEEDCS17E * RATIOHL
      WSDELCS17E = AMIN1(WSALLCS17E,CAPACCS17E)
C
      IF (WSALLCS17E .GT. 0.0) THEN
         RATIOWS = WSDELCS17E / WSALLCS17E
      ELSE
         RATIOWS = 1.0
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      QU(IES) = QU(IES) + WSNDES * RATIOWS * RATIOHL * 86400.
      QU(IESE)= QU(IESE)+ WSNDESE * RATIOWS * RATIOHL * 86400.
      rlocal_ws_to_es = RATIOWS * RATIOHL*WSNDES * 86400.
      rlocal_ws_to_ese = RATIOWS * RATIOHL*WSNDESE * 86400.
      WSNDES = (1.0 - RATIOWS * RATIOHL) * WSNDES
      WSNDESE = (1.0 - RATIOWS*RATIOHL) * WSNDESE
      DQU(IES) = WSNDES
      iup_cnl_ies = iup_canal_ws(ies)
      if (iup_cnl_ies .le. nch) then
        DQU(iup_cnl_ies) = DQU(iup_cnl_ies) - rlocal_ws_to_es/86400.
        DQU(iup_cnl_ies) = AMAX1(DQU(iup_cnl_ies),0.0)
      endif
      KFLO(IFLCS17E) = WSDELCS17E
      IF (runmode .EQ. 'CALIB') THEN
         istr_count = istr_count + 1
         istr_index(istr_count) = IFLCS17E
         JSTR = JSTR + 1
         STRFLW(JSTR,IDSS) = KFLO(IFLCS17E)
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C      JSTR = JSTR + 1
C      STRFLW(JSTR,IDSS) = KFLO(IFLCS17E)
C
      RETURN
      END
```

lok_bndry_input_data.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lok_bndry_input_data.F,v 1.22 2004/02/05 18:59:29 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lok_bndry_input_data.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO OPEN I/O FILES AND RETRIEVE PHYSICAL DATA REQUIRED BY MODEL
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lok_bndry_input_data (imth, ibm, ifyr, iyear,
     & climate_index, no_of_hurr_pred, rmulti_seas_pred,
     & iatl_convyr_index, istr_index, kflo0, mod_delta_stor, s65ernff,
     & tcnsrnff, dmndro, stltrro, s4runff, bmp_makeup_water, fplresws,
     & dmdsemind, srsdmnd, srsdmnd_w, srsdmnd_e, sle_dmd_day,
     & cale_dmd_day, c43rf, c44rf, norstorrf, sem_cyp_dmnd,
     & diff_runoff_bmp, frac_area_eaa_basin, isem_flg, lp, nday_yr,
     & ro_factor, dmdronm, numd, morefet, lokmorn, ccdmdro, cfcurfut,
     & rf_factor, rlosa_factor, itribres2, idbsnopt1, idbsnopt2,
     & sle_dmnd_factor, cale_dmnd_factor, inorth_stor_opt, frac_sem_cyp,
     & bmp_opt, nflpts, ifltab1, ifltabr, pdsi, diston_capac,
     & diston_frac, diston_to_cal, diston_from_cal, diston_to_s4,
     & diston_from_s4, caldmndro, s4dmndro, nlsdmndro, tcnsro,
     & semirr_dmnd_factor, fpl_dmnd_factor, sem_cyp_tot_dmnd,
     & sem_cyp_dmnd_local,uisrnff)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'

      DIMENSION bmp_makeup_water(366), c43rf(366), c44rf(366),
     & cale_dmd_day(366), climate_index(12),
     & diff_runoff_bmp(max_n_eaa_condt,366), dmdsemind(366),
```

```
     & dmndro(8,2,2,366), fplresws(366),
     & frac_area_eaa_basin(max_n_eaa_condt), ifltab1(1200),
     & ifltabr(1200), istr_index(max_n_str_node_dep),
     & kflo0(max_n_struc_flw,366), lokmorn(12), lp(60),
     & mod_delta_stor(366), morefet(12), norstorrf(366), numd(12),
     & pdsi(12), rf_factor(12), rlosa_factor(8,2), rmulti_seas_pred(12),
     & ro_factor(12), s4runff(366), s65ernff(366), sem_cyp_dmnd(366),
     & sle_dmd_day(366), srsdmnd(366), srsdmnd_e(366), srsdmnd_w(366),
     & stltrro(366), tcnsrnff(366), tmpvals(366), diston_capac(2),
     & diston_frac(2), diston_to_cal(366), diston_from_cal(366),
     & diston_to_s4(366), diston_from_s4(366), caldmndro(2,366),
     & s4dmndro(2,366), nlsdmndro(2,366), tcnsro(366),
     & sem_cyp_tot_dmnd(366), sem_cyp_dmnd_local(366), uisrnff(366)

      CHARACTER bmp_opt*5, ca*5, cc*4, ccdmdro(2)*6, ce*4, cf*20,
     & cfcurfut(2)*7, cftrib*10, cpath*80, ctype*8, cunits*3,
     & dmdronm(10)*10, sdate*9, stime*4

      LOGICAL lp

      REAL kflo0,lokmorn,mod_delta_stor,morefet,norstorrf,nlsdmndro,
     & tmptcns

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

       ca = 'SFWMM'
       ce = '1DAY'
       cc = 'FLOW'
       stime = '1200'
       iyr = ifyr+iyear-1

       if (IMTH.EQ.IBM) then
         nday_yr = iymdjl(ifyr,ibm,1) - iymdjl(ifyr,1,1)
       else
         nday_yr = 0
       endif
c
c  write header to tape62 - file which contains capacity information
c    for key structures conveying water supply releases from LOK/WCA
c    to LEC Service areas
c
        if (lp(23)) then
         write(62,155) iyear,ifyr+iyear-1
 155     format(2(/),10X,'YEAR OF SIMULATION',i5,5x,'RAINFALL YEAR'
     $    ,i10)
        endif
c
c  input climate variables for WSE schedule for LOK
c   and WCA operations
c
        read (108,'(4x,i3)') no_of_hurr_pred
        read (109,'(4X,12F8.2)') (climate_index
     $     (i),i = 1,12)
        read (135,'(4X,12f6.2,2X,I1)') (rmulti_seas_pred(i),
     $     i=1,12),iatl_convyr_index
        read (148,'(8X,12F8.1)') (pdsi(i),i=1,12)
c
```

```
c  read input from various dss files
c
c note: will only work if simulation starts on a january
c read one year's worth of data at the beginning of the year
c calculate number of days to read for this year
            nperiods = iymdjl(iyr,12,31) - iymdjl(iyr,1,1) + 1
c convert year,month,day integers to character format
            call ymddat(iyr,1,1,104,sdate,nd,ierror)
c do one dss read per structure ( one read = 365 or 366 daily values)
            do i=1,nflpts
             is = istr_index(i)
c form pathname
         if (flnm(is) .eq. 'G155  ' .or. flnm(is) .eq. 'G136  ' .or.
     &       flnm(is) .eq. 'G88   ' .or. flnm(is) .eq. 'G89   '
     &       .or. flnm(is) .eq. 'G155SP') then
            cf = 'BASE'
         elseif(flnm(is) .eq. 'S190  ') then
            if (isem_flg .eq. 1) then
             cf = 'EST_HIST_V5.0_REV'
            else
             cf = 'EST_HIST_V5.0'
            endif
         else
           cf = 'HISTORICAL'
         endif
               call zpath(ca,flnm(is),cc,sdate,ce,cf,cpath,npath)
c read from dss file
               call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &         tmpvals,cunits,ctype,iofset,istat)
               if(istat.ne.0)then
                 write(*,*) 'Error reading structure flow for ',flnm(is)
                 write(*,*) 'ISTAT:',istat
                 write(*,*) 'Consult HECDSS manual!!'
                 stop
               endif
c convert read values to integer values
               do j = 1,nperiods
                  kflo0(is,j) = tmpvals(j)
                  if (kflo0(is,j).eq.-901. .or. kflo0(is,j).eq.-902.)
     &             kflo0(is,j) = 0.0
               enddo
            enddo
c
c read Modified Delta Storage for LOK in flo.dss
c
        call zpath(ca,'MDS','FLOW',sdate,ce,
     &                     'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             mod_delta_stor(l) = tmpvals(l)
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for MDS'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
```

```
            stop
         endif
c
c read S-65E from flo.dss
c
c
C  read in estimated flows thru S-65E into Lake Okeechobee
c
       if (typerun .ne. 'FUTURE ') then
        call zpath(ca,'S65E','FLOW',sdate,ce,
     &                    'CURR_REV',cpath,npath)
       else
        call zpath(ca,'S65EFR','FLOW',sdate,ce,
     &                    'FUT_REV',cpath,npath)
       endif
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             s65ernff(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for S65E'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c
c read Taylor Creek/Nubbin Slough Flows from flo.dss
c
        call zpath(ca,'TCNSQ','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             tcnsro(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for TCNSQ'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c
c read Upstream Taylor Creek/Nubbin Slough Flows from flo.dss
c
        call zpath(ca,'TCNSQ','FLOW',sdate,ce,
     &                    'EST_HIST_V5.0_REV',cpath,npath)
c read from the same flo.dss file
         call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             tcnsrnff(l) = tmpvals(l)*1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for TCNSQ'
```

```
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
          endif
c
c  read in estimated flows from Upper Istokpoga into Lake Okeechobee
c    (from S70+S75 via S71+S72)
c
        if (typerun .ne. 'FUTURE ') then
         call zpath(ca,'UISTLK','FLOW',sdate,ce,
     &                    'CURRENT',cpath,npath)
        else
         call zpath(ca,'UISTLK','FLOW',sdate,ce,
     &                    'FUTURE',cpath,npath)
        endif
c read from the same flo.dss file
          call zrrts(ifltabr,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
            do l = 1,nperiods
              uisrnff(l) = tmpvals(l)*1.9835
            enddo
          if(istat.ne.0)then
            write(*,*) 'Error reading flo.dss flow for UISTLK'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
          endif
c
cc define number of runoff and demand points input
c convert year,month,day integers to character format
       call ymddat(iyr,1,1,104,sdate,nd,ierror)
cv>>
        ndmdropts = 8
cv<<
       do i=1,ndmdropts
        do j=1,2
c this loop forms pathname for demand/runoff
         if (demand_level_opt .eq. 'CURR') then
           k = 1
         else
           k = 2
         endif
         call zpath(ca,dmdronm(i),ccdmdro(j),sdate,ce,
     &                        cfcurfut(k),cpath,npath)
cc do one dss read per basin ( one read = 365 or 366 daily values)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &             tmpvals,cunits,ctype,iofset,istat)
         if(istat.ne.0)then
           write(*,*) 'Error reading dmd/ro flow for ',dmdronm(i)
           write(*,*) 'ISTAT:',istat
           write(*,*) 'Consult HECDSS manual!!'
           stop
         endif
c convert read values to integer values . is this reqd??
         do l = 1,nperiods
           dmndro(i,j,k,l) = tmpvals(l) * rlosa_factor(i,j)
         enddo
```

```
       enddo
      enddo
      IF (simsles .ne. 'NO ') THEN
        if(itribres2 .eq. 1) then
          cftrib = 'FUTRES'
        else
          if (demand_level_opt .eq. 'CURR') then
            cftrib = 'CURRENT'
          else
            cftrib = 'FUTURE'
          endif
        endif
clmbasr<<
c  read in st lucie tributary flows
         call zpath(ca,'SLTRIB','RUNOFF',sdate,ce,
     &                cftrib,cpath,npath)
c  read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             stltrro(l) = tmpvals(l)
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SLTRIB'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
       ENDIF

c
C  read in measured s4 basin outflows at s235
c
c       IF (runmode .EQ. 'SIMUL') THEN
c        call zpath(ca,'S4BSN','RUNOFF',sdate,ce,
c     &                    'CURRENT',cpath,npath)
c
c read from the same dmd/ro dss file
c
c         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
c     &                 tmpvals,cunits,ctype,iofset,istat)
c           do l = 1,nperiods
c             s4runff(l) = tmpvals(l)
c           enddo
c         if(istat.ne.0)then
c            write(*,*) 'irror reading dmd/ro flow for S4BSN'
c            write(*,*) 'ISTAT:',istat
c            write(*,*) 'Consult HECDSS manual!!'
c            stop
c         endif
c      ENDIF
c
c read in BMP Makeup Water according to BMP Makeup Rule
c
        if (make_up_water_opt .eq. 'MAKEUP ') then
          call zpath(ca,'BMP','REPLWATER',sdate,ce,
```

```
     &                     'RULE',cpath,npath)
c
c read from the same dmd/ro dss file
c
          call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             bmp_makeup_water(l) = tmpvals(l)
           enddo
          if(istat.ne.0)then
            write(*,*) 'irror reading dmd/ro flow for BMPMAKEUP'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
          endif
        endif
c
c  read in Flows to FPL Reservoir from LOK
c
       if (demand_level_opt .eq. 'CURR') then
         call zpath(ca,'FPLRES','FLOW',sdate,ce,
     &                     'CURRENT',cpath,npath)
       else
         call zpath(ca,'FPLRES','FLOW',sdate,ce,
     &                     'FUTURE',cpath,npath)
       endif
c read from the same flo.dss file
       call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
        do l = 1,nperiods
          fplresws(l) = tmpvals(l) * fpl_dmnd_factor
        enddo
        if(istat.ne.0)then
           write(*,*) 'Error reading flo.dss flow for FPLRES'
           write(*,*) 'ISTAT:',istat
           write(*,*) 'Consult HECDSS manual!!'
           stop
        endif
c
c read in estimated demands of Seminole indians on north side of
c Lake Okeechobee
c
        if (demand_level_opt .eq. 'CURR') then
          call zpath(ca,'SEMIRR','DEMAND',sdate,ce,
     &                     'CURRENT',cpath,npath)
        else
          call zpath(ca,'SEMIRR','DEMAND',sdate,ce,
     &                     'FUTURE',cpath,npath)
        endif
c        call zpath(ca,'SEMIRR','DEMAND',sdate,ce,
c     &                      'CURRENT',cpath,npath)
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             dmdsemind(l) = tmpvals(l) * semirr_dmnd_factor
           enddo
```

```
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SEMIRR'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
cc if read-in option true, do one last dss read for rainfall-driven
cc Shark River Slough current base Demands.
      if (type_flow_across_ttrail .ne. 'MINDEL' .AND. opt_ft_input
     &  .and. type_flow_across_ttrail .ne. 'STAGETG') then
        if (type_target .eq. 'TOTAL' .or. type_flow_across_ttrail
     &           .eq. 'RFPLAN ') then
          if (type_flow_across_ttrail .eq. 'NSMFLOW') then
            call zpath(ca,'SRS','DEMAND',sdate,ce,
     &                     'FUTURE',cpath,npath)
          else
            call zpath(ca,'SRS','DEMAND',sdate,ce,
     &                     'CURRENT',cpath,npath)
          endif
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             srsdmnd(l) = tmpvals(l)/1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SRS'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
        else
          if (type_flow_across_ttrail .eq. 'NSMFLOW') then
            call zpath(ca,'SRS','DMD_W',sdate,ce,
     &             'FUTURE',cpath,npath)
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             srsdmnd_w(l) = tmpvals(l)/1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SRS'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
            call zpath(ca,'SRS','DMD_E',sdate,ce,
     &             'FUTURE',cpath,npath)
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
           do l = 1,nperiods
             srsdmnd_e(l) = tmpvals(l)/1.9835
           enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SRS'
```

```
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
        endif
        endif
      endif
clmb>>
c read estuarine demands
c
c read daily estuarine demands
        if (simsles .ne. 'NO ') then
c for St. Lucie
         if (demand_level_opt .eq. 'CURR') then
           call zpath(ca,'STL_ESTU','DEMAND',sdate,ce,
     &                     'CURRENT',cpath,npath)
         else
           call zpath(ca,'STL_ESTU','DEMAND',sdate,ce,
     &                     'FUTURE',cpath,npath)
         endif
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            sle_dmd_day(l) = tmpvals(l) * sle_dmnd_factor
         enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for STL_ESTU'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
        endif
c
               if (simcaes .ne. 'NO ') then
c for Caloosahatchee
               if (demand_level_opt .eq. 'CURR') then
                 call zpath(ca,'CAL_ESTU','DEMAND',sdate,ce,
     &                     'CURRENT',cpath,npath)
               else
                 call zpath(ca,'CAL_ESTU','DEMAND',sdate,ce,
     &                     'FUTURE',cpath,npath)
               endif
c              endif
c read from the same dmd/ro dss file
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            cale_dmd_day(l) = tmpvals(l) * cale_dmnd_factor
         enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for CAL_ESTU'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
               endif
```

```
c
c read this year's daily c-43/c-44 basin rainfall
c to be applied to detention basin/reservoir option
      if(idbsnopt1.eq.1) then
         call zrrts (ifltab1,'/SFWMM/C43/RF//1DAY/HISTORICAL/',
     &      sdate,stime,nperiods,tmpvals,cunits,ctype,iofset,istat)
         if(istat.ne.0)then
            print *,'Error reading c-43 basin rainfall data'
            print *,'ISTAT:',istat
            print *,'Consult HECDSS manual!!'
         endif
c convert inches to feet
         do l = 1,nperiods
            c43rf(l) = tmpvals(l)/12.
         enddo
      endif
      if(idbsnopt2.eq.1) then
         call zrrts (ifltab1,'/SFWMM/C44/RF//1DAY/HISTORICAL/',
     &      sdate,stime,nperiods,tmpvals,cunits,ctype,iofset,istat)
         if(istat.ne.0)then
            print *,'Error reading c-44 basin rainfall data'
            print *,'ISTAT:',istat
            print *,'Consult HECDSS manual!!'
            stop
         endif
c convert inches to feet
         do l = 1,nperiods
            c44rf(l) = tmpvals(l)/12.
         enddo
      endif
c
c
c read this year's daily North storage reservoir rainfall
c to be applied to detention basin/reservoir option
      if(inorth_stor_opt .eq. 1) then
         call zrrts (ifltab1,'/SFWMM/NORLOK/RF//1DAY/HISTORICAL/',
     &      sdate,stime,nperiods,tmpvals,cunits,ctype,iofset,istat)
         if(istat.ne.0)then
            print *,'Error reading North storage basin rainfall data'
            print *,'ISTAT:',istat
            print *,'Consult HECDSS manual!!'
            stop
         endif
c convert inches to feet
         do l = 1,nperiods
            norstorrf(l) = tmpvals(l)/12.
         enddo
      endif
c
c read seminole indians agric demands in big cypress
c

      if (isem_flg .eq. 1) then
         call zpath(ca,'SEM_CYPD','DEMAND',sdate,ce,
     &                    'SUPP',cpath,npath)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
```

```
         do l = 1,nperiods
            sem_cyp_dmnd(l) = tmpvals(l)
         enddo
         call zpath(ca,'SEM_CYPD','DEMAND',sdate,ce,
     &                    'TOTAL',cpath,npath)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            sem_cyp_dmnd_local(l) = tmpvals(l) - sem_cyp_dmnd(l)
            sem_cyp_dmnd(l) = sem_cyp_dmnd(l) * frac_sem_cyp
            sem_cyp_tot_dmnd(l) = sem_cyp_dmnd(l) +
     &        sem_cyp_dmnd_local(l)
         enddo

         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for SEM_CYPD'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
       endif
c
c read runoff reduction due to BMPs for major EAA basins
c
      if (bmp_opt .eq. 'TRUE ') then
c Miami Basin
         call zpath(ca,'EAAMIAMIB','RUNOFF',sdate,ce,
     &                    'BMPRED',cpath,npath)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            diff_runoff_bmp(1,l) = tmpvals(l)
     &                * frac_area_eaa_basin(1)
         enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for MIAMIBMPRED'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c NNR_HIL Basin
         call zpath(ca,'EAANNRHIL','RUNOFF',sdate,ce,
     &                    'BMPRED',cpath,npath)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            diff_runoff_bmp(2,l) = tmpvals(l)
     &                * frac_area_eaa_basin(2)
         enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for NNRHILMPRED'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
c WPB Canal Basin
         call zpath(ca,'EAAWPB','RUNOFF',sdate,ce,
```

```
     &                    'BMPRED',cpath,npath)
         call zrrts(ifltab1,cpath(1:npath),sdate,stime,nperiods,
     &                 tmpvals,cunits,ctype,iofset,istat)
         do l = 1,nperiods
            diff_runoff_bmp(3,l) = tmpvals(l) *
     &           frac_area_eaa_basin(3)
         enddo
         if(istat.ne.0)then
            write(*,*) 'Error reading dmd/ro flow for WPBBMPRED'
            write(*,*) 'ISTAT:',istat
            write(*,*) 'Consult HECDSS manual!!'
            stop
         endif
       endif
c
c   adjust boundary flows/runoff for increase/decrease in rainfall
c
      idayno_r = 0
      numd(2) = 28
      if(mod(ifyr+iyear-1,4) .eq. 0) numd(2) = 29
      do imo = 1,12
        idayno_r = idayno_r + 1
        stltrro(idayno_r) = stltrro(idayno_r) * ro_factor(imo)
        if(idbsnopt1.eq.1) c43rf(idayno_r) = c43rf(idayno_r)
     $      * rf_factor(imo)
        if(idbsnopt2.eq.1) c44rf(idayno_r) = c44rf(idayno_r)
     $      * rf_factor(imo)
        if(inorth_stor_opt .eq. 1) norstorrf(idayno_r)
     $      = norstorrf(idayno_r) * rf_factor(imo)
        tcnsrnff(idayno_r) = tcnsrnff(idayno_r) * ro_factor(imo)
        s65ernff(idayno_r) = s65ernff(idayno_r) * ro_factor(imo)
        uisrnff(idayno_r) = uisrnff(idayno_r) * ro_factor(imo)
c        s4runff(idayno_r) = s4runff(idayno_r) * ro_factor(imo)
        do ifl=1,nflpts
          is = istr_index(ifl)
          kflo0(is,idayno_r) = kflo0(is,idayno_r) * ro_factor(imo)
        enddo
c        do i=1,ndmdropts
c          if (demand_level_opt .eq. 'CURR') then
c            k = 1
c          else
c            k = 2
c          endif
c          dmndro(i,1,k,idayno_r) = dmndro(i,1,k,idayno_r)
c     $         * ro_factor(imo)
c        enddo
      enddo
C-------------------------------------------------------
C     READ ONE YEAR OF MONTHLY ET AND RAINFALL DATA FOR
C     LAKE OKEECHOBEE
C-------------------------------------------------------
      DO  KMONTH = 1,12
        READ(36,'(6X,5F10.0)') MOREFET(KMONTH), LOKMORN(KMONTH)
      ENDDO


C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
        if (demand_level_opt .eq. 'CURR') then
          k = 1
        else
          k = 2
        endif
        do l = 1,nperiods
C-----------------------------------------------------
C      Adjust S4 Other and Caloos Basins by effect of
c      S4 DISTON
C-----------------------------------------------------
            do j=1,2
                caldmndro(j,l) = dmndro(2,j,k,l)
                s4dmndro(j,l) = dmndro(5,j,k,l)
            enddo

            diston_to_cal(l) = min(dmndro(4,1,k,l)*diston_frac(1),
     $          diston_capac(1)*1.9835)
            diston_from_cal(l) = min(dmndro(4,2,k,l)*diston_frac(2),
     $          diston_capac(2)*1.9835)
            diston_to_s4(l) = dmndro(4,1,k,l) - diston_to_cal(l)
            diston_from_s4(l) = dmndro(4,2,k,l) - diston_from_cal(l)

            dmndro(2,1,k,l) = dmndro(2,1,k,l) + diston_to_cal(l)
            dmndro(2,2,k,l) = dmndro(2,2,k,l) + diston_from_cal(l)
            IF ((dmndro(2,1,k,l) - dmndro(2,2,k,l)) .ge. 0.0) THEN
                 dmndro(2,1,k,l) = (dmndro(2,1,k,l) -
     $               dmndro(2,2,k,l))
                 dmndro(2,2,k,l) = 0.0
            ELSE
                 dmndro(2,2,k,l) = (dmndro(2,2,k,l) -
     $               dmndro(2,1,k,l))
                 dmndro(2,1,k,l) = 0.0
            ENDIF

            dmndro(5,1,k,l) = dmndro(5,1,k,l) + diston_to_s4(l)
            dmndro(5,2,k,l) = dmndro(5,2,k,l) + diston_from_s4(l)
            IF ((dmndro(5,1,k,l) - dmndro(5,2,k,l)) .ge. 0.0) THEN
                 dmndro(5,1,k,l) = (dmndro(5,1,k,l) -
     $               dmndro(5,2,k,l))
                 dmndro(5,2,k,l) = 0.0
            ELSE
                 dmndro(5,2,k,l) = (dmndro(5,2,k,l) -
     $               dmndro(5,1,k,l))
                 dmndro(5,1,k,l) = 0.0
            ENDIF
            s4runff(l) = dmndro(5,1,k,l)

C-----------------------------------------------------
C      Try to meet NLS demand with TCNSQ
C-----------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            do j=1,2
                nlsdmndro(j,l) = dmndro(6,j,k,l)
            enddo

            IF (dmndro(6,2,k,l) .gt. 0.0) THEN
               IF ((dmndro(6,2,k,l) - tcnsrnff(l)) .ge. 0.0) THEN
```

```
            dmndro(6,2,k,l) = (dmndro(6,2,k,l) - tcnsrnff(l))
            tcnsrnff(l) = 0.0
         ELSE
            tcnsrnff(l) = (tcnsrnff(l) - dmndro(6,2,k,l))
            dmndro(6,2,k,l) = 0.0
         ENDIF
      ELSE
         tmptcns = tcnsrnff(l) + dmndro(6,1,k,l)
         tcnsrnff(l) = amin1(tcnsro(l),tmptcns)
         dmndro(6,1,k,l) = tmptcns - tcnsrnff(l)
      ENDIF
  enddo

return
end
```

lok_o_wca_in_struc_dta.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: lok_o_wca_in_struc_dta.F,v 1.15 2004/02/09 21:32:43 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lok_o_wca_in_struc_dta.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO OPEN I/O FILES AND RETRIEVE PHYSICAL DATA
C                  REQUIRED BY MODEL
C   FILES OPENED:
C    BRF6589   DAILY RAINFALL DATA, BY BASIN, FOR SIMULATION PERIOD
C    WELLDAT   AVE. DAILY PUMPAGE,BY MONTH,FOR EACH WELL (MGD)(HISTORICAL)
C    WELPRDT   AVE. DAILY PUMPAGE,BY MONTH,FOR EACH WELL (MGD)(SIMULATION)
C    FLO2X2    DAILY FLOWS FOR EACH KNOWN FLOW POINT (CFS)
C    LECDEF    RUN DEFINITION & MISCELLANEOUS DATA
C
C   INPUT VARIABLES USED:
C    ELLS()    TOPOGRAPHY INFORMATION FOR EACH NODE POINT (FT NGVD)
C    S()       STORAGE COEFFICIENT (DIMENSIONLESS)
C    LUTYP()   LAND USE INDICATORS FOR EACH NODE
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lok_o_wca_in_struc_dta (ires_no, idshg_opt, noutlets,
     & istrctr_ws, nres_ws, istr_count, struc_name_meas)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
```

```
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'add_misc.inc'
      INCLUDE 'eaacap.inc'
      INCLUDE 'STRCOUNT.inc'

      DIMENSION icol_add(max_n_str_lok_in_wca),
     & idshg_opt(max_n_res,max_nostr_res),
     & ihw_col(max_n_str_lok_in_wca), ihw_row(max_n_str_lok_in_wca),
     & ires_no(max_n_res), irow_add(max_n_str_lok_in_wca),
     & istrctr_ws(max_n_res,max_nostr_res),
     & itw_col(max_n_str_lok_in_wca), itw_row(max_n_str_lok_in_wca),
     & noutlets(max_n_res)

      CHARACTER add_can_dep_id(10)*5, add_data_need_opt*6,
     & add_str_dep_id(10)*6, cictsta*6,
     & idn_canal_name(max_n_str_lok_in_wca)*5,
     & iup_canal_name(max_n_str_lok_in_wca)*5,
     & struc_name_meas(max_n_struc_meas)*6,
     & struc_name_sim(max_n_str_lok_in_wca)*6

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  account for local lwdd structures:calibration run only

      ILWDDP = 0
      IF (runmode .EQ. 'CALIB') THEN
         ILWDDP = ILWDDP + 1
         CALL match('CS2   ',1,6,flnm,300,6,iflcs2)
         istr_INDEX(nflpts+ilwddp) = iflcs2
         STRNAME(ILWDDP) = FLNM(iflcs2)
         ILWDDP = ILWDDP + 1
         CALL match('CS9   ',1,6,flnm,300,6,iflcs9)
         istr_INDEX(nflpts+ilwddp) = iflcs9
         STRNAME(ILWDDP) = FLNM(iflcs9)
         ILWDDP = ILWDDP + 1
         CALL match('CS12  ',1,6,flnm,300,6,iflcs12)
         istr_INDEX(nflpts+ilwddp) = iflcs12
         STRNAME(ILWDDP) = FLNM(iflcs12)
         ILWDDP = ILWDDP + 1
         CALL match('CS17W ',1,6,flnm,300,6,iflcs17w)
         istr_INDEX(nflpts+ilwddp) = iflcs17w
         STRNAME(ILWDDP) = FLNM(iflcs17w)
         ILWDDP = ILWDDP + 1
         CALL match('CS17E ',1,6,flnm,300,6,iflcs17e)
         istr_INDEX(nflpts+ilwddp) = iflcs17e
         STRNAME(ILWDDP) = FLNM(iflcs17e)
      ENDIF
c
c  input structures to be simulated explicitly in route subroutine
c
      READ (101,*) ncalcpt
      READ (101,'(I3,2x,30(A6,1x))') no_struc_spec_code ,
     $(struc_name_spec_code(i),i = 1,no_struc_spec_code)
      DO INDEX = 1,ncalcpt
```

```
         READ (101,'(3(A6,1x))') struc_name_sim(INDEX) ,
     $   add_data_need_opt,CICTSTA
         IF (add_data_need_opt .EQ. 'DATA  ') THEN
            READ (101,*) ICODE(INDEX),dischg_c(INDEX) ,expon(INDEX),
     $      type_flow_s(INDEX)
            READ (101,*) ihw_opt(INDEX)
            IF (ihw_opt(INDEX) .EQ. 1) THEN
               READ (101,*) ihw_col(INDEX),ihw_row(INDEX)
            ELSE
               READ (101,'(A5)') iup_canal_name(INDEX)
            ENDIF
            READ (101,*) itw_opt(INDEX)
            IF (itw_opt(INDEX) .EQ. 1) THEN
               READ (101,*) itw_col(INDEX),itw_row(INDEX)
            ELSE
               READ (101,'(A5)') idn_canal_name(INDEX)
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            READ (101,*) n_bkpts,(stg_bkpt(INDEX,i),i=1,n_bkpts)
            READ (101,*) n_cbkpts,(cstg_bkpt(INDEX,i),i=1,n_cbkpts)
            READ (101,*) n_add_can_dep(INDEX),(add_can_dep_id(i) ,i=1,
     $      n_add_can_dep(INDEX))
            READ (101,*) n_add_grid_loc(INDEX),(icol_add(i),irow_add
     $      (i) ,i=1,n_add_grid_loc(INDEX))
            READ (101,*) n_add_str_dep(INDEX),(add_str_dep_id(i) ,i=1,
     $      n_add_str_dep(INDEX))
c
            DO i = 1,n_add_can_dep(INDEX)
               CALL match(add_can_dep_id(i),1,5,cnm,nch+1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,10) add_can_dep_id(i),struc_name_sim(INDEX)
                  STOP
               ENDIF
               canal_no_add(INDEX,i) = imatch
            END DO
            DO i = 1,n_add_str_dep(INDEX)
               CALL match(add_str_dep_id(i),1,6,flnm,300,6,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,20) add_str_dep_id(i),struc_name_sim(INDEX)
                  STOP
               ENDIF
               add_dep_str_indx(INDEX,i) = imatch
            END DO
            DO i = 1,n_add_grid_loc(INDEX)
               node_add(INDEX,i) = icol_add(i) - minx(irow_add(i)) +  1+
     $         isum(irow_add(i))
            END DO
            IF (ihw_opt(INDEX) .EQ. 0) THEN
               CALL match(iup_canal_name(INDEX),1,5,cnm,nch+1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,30) iup_canal_name(INDEX),
     $            struc_name_sim(INDEX)
                  STOP
               ENDIF
               iup_canal_INDEX(INDEX) = imatch
            ELSE
               node_hw(INDEX) = ihw_col(INDEX) - minx(ihw_row(INDEX)) +
```

```
     $          1+ isum(ihw_row(INDEX))
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (itw_opt(INDEX) .EQ. 0) THEN
               CALL match(idn_canal_name(INDEX),1,5,cnm,nch+1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,30) idn_canal_name(INDEX)
                  STOP
               ENDIF
               idn_canal_INDEX(INDEX) = imatch
            ELSE
               node_tw(INDEX) = itw_col(INDEX) - minx(itw_row(INDEX)) +
     $          1+ isum(itw_row(INDEX))
            ENDIF
         ELSE
            icode(INDEX) = 'SPC'
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
         CALL match(struc_name_sim(INDEX),1,6,flnm,300,6,imatch)
         IF (imatch .EQ. 0) THEN
            WRITE(*,40) struc_name_sim(INDEX)
            STOP
         ENDIF
         icalcpt(INDEX) = imatch
         istr_INDEX(nflpts + INDEX + ilwddp) = imatch
         CALL match(CICTSTA,1,5,resname,ntotres + 1,5,imatch)
         IF (imatch .NE. 0) THEN
            ictsta(INDEX) = imatch
         ELSE
            ictsta(INDEX) = 99
         ENDIF
         IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(11)) THEN
            is8_no_sim = INDEX
            s8max = dischg_c(INDEX)
         ELSE IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(10))
     $   THEN
            is7_no_sim = INDEX
            s7max = dischg_c(INDEX)
         ELSE IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(8))
     $   THEN
            is6_no_sim = INDEX
            s6max = dischg_c(INDEX)
         ELSE IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(7))
     $   THEN
            is5a_no_sim = INDEX
            s5amax = dischg_c(INDEX)
         ELSE IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(9))
     $   THEN
            is150_no_sim = INDEX
         ELSE IF (struc_name_sim(INDEX) .EQ. struc_name_spec_code(21))
     $   THEN
            s140cap = dischg_c(INDEX)
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
      END DO
      s8max = s8max * RCPFACTEAA(1)
      s7max = s7max * RCPFACTEAA(2)
```

```
      s6max = s6max * RCPFACTEAA(4)
      s5amax = s5amax * RCPFACTEAA(3)
C
C STRUCTURE NAMES HANDLED IN ROUTE TO STR2x2.DSS OUTPUT FILE
C
      IS1211=0
      IF (runmode .EQ. 'CALIB') THEN
         DO I = 1,NFLPTS
           call match(struc_name_meas(i),1,6,flnm,300,6,imatch_s)
           IF (flnm(imatch_s) .eq. 'S11   ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S11   '
           ELSEIF (flnm(imatch_s) .eq. 'S337  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S337  '
           ELSEIF (flnm(imatch_s) .eq. 'S339  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S339  '
           ELSEIF (flnm(imatch_s) .eq. 'S340  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S340  '
           ELSEIF (flnm(imatch_s) .eq. 'S140  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S140  '
           ELSEIF (flnm(imatch_s) .eq. 'S175  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S175  '
           ELSE IF (flnm(imatch_s).eq.'S197  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S197  '
           ELSEIF (flnm(imatch_s) .eq. 'S10   ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S10   '
           ELSEIF (flnm(imatch_s) .eq. 'S332  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S332  '
           ELSEIF (flnm(imatch_s) .eq. 'S38   ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S38   '
           ELSEIF (flnm(imatch_s) .eq. 'S9    ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S9    '
           ELSEIF (flnm(imatch_s) .eq. 'S5A3  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S5A2  '
           ELSEIF (flnm(imatch_s) .eq. 'S5A4  ') THEN
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = flnm(imatch_s)
              IS1211 = IS1211 + 1
              STRNAME(IS1211+ILWDDP) = 'S5A3  '
           ENDIF
         ENDDO
      ENDIF
      JS = 0
      DO I = 1,nres_ws
         DO IS = 1,noutlets(i)
            ks = istrctr_ws(ires_no(i),is)
```

```
            if (res_out_strname(ires_no(i),ks) .eq.
     $      res_struc_name_spec_code(15)) go to 50
            IF (idshg_opt(ires_no(i),is) .NE. 1 .AND. idshg_opt(ires_no
     $      (i),is) .NE.  3) THEN
               JS = JS + 1
               STRNAME(JS+IS1211+ILWDDP) = res_out_strname(ires_no(i) ,
     $         ks)
            ENDIF
   50       CONTINUE
         END DO
      END DO
C
      DO i = 1,nres_ws
         DO is = 1,noutlets(i)
            ks = istrctr_ws(ires_no(i),is)
            IF (idshg_opt(ires_no(i),is) .EQ. 3) THEN
               JS = JS + 1
               STRNAME(JS+IS1211+ILWDDP) = res_out_strname(ires_no
     $         (i) ,ks)
            ENDIF
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
      IF (opt_for_hill_bypass .EQ. 'FALSE') THEN
         DO I = 1,nres_ws
            DO IS = 1,noutlets(i)
               IF (idshg_opt(ires_no(i),is) .NE. 1 .AND. idshg_opt
     $         (ires_no(i),is) .NE. 3) THEN
                  IF (res_out_strname(ires_no(i),istrctr_ws(ires_no(i),
     $            is)) .NE. res_struc_name_spec_code(15)) GO TO 60
                  JS = JS + 1
                  STRNAME(JS+IS1211+ILWDDP) = res_out_strname(ires_no
     $            (i) ,istrctr_ws(ires_no(i),is))
   60             CONTINUE
               ENDIF
            END DO
         END DO
      ENDIF
      istr_count = nflpts + ILWDDP
      DO 70 I=1,NCALCPT
         STRNAME(I+IS1211+JS+ILWDDP) = FLNM(ICALCPT(I))
         istr_count = istr_count + 1
   70 CONTINUE
      MAXSTR=NCALCPT + IS1211 + JS + ILWDDP
      IF (runmode .EQ. 'SIMUL') THEN
         MAXSTR=MAXSTR+1
         STRNAME(MAXSTR) = FLNM(ifls77)
         MAXSTR=MAXSTR+1
         STRNAME(MAXSTR) = FLNM(ifls308)
      ENDIF
      RETURN
   10 FORMAT(2(/),'UNRECOGNIZED ADD CANAL DEP ID',2x,A5,2x,
     $'FOR STRUCTURE',2x,A6,2x,'CHECK gen_nodal_dep_struc.dat FILE')
   20 FORMAT(2(/),'UNRECOGNIZED ADD STRUC DEP ID',2x,A6,2x,
     $'FOR STRUCTURE',2x,A6,2x,'CHECK gen_nodal_dep_struc.dat FILE')
   30 FORMAT(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,2x,
```

```
      $'FOR STRUCTURE',2x,A6,2x,'CHECK gen_nodal_dep_struc.dat FILE')
   40 FORMAT(2(/),'UNRECOGNIZED STRUC',2x,A6,2x,
      $'CHECK gen_nodal_dep_struc.dat FILE')
       END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

lok_ws_to_larger_res.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lok_ws_to_larger_res.F,v 1.10 2005/06/10 13:29:11 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lok_ws_to_larger_res.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE COMPUTES WATER SUPPLY DELIVERIES FROM LAKE
C   OKEECHOBEE TO MAINTAIN MINIMUM DEPTHS OF WATER IN THE STORMWATER
C   TREATMENT AREAS (STAs) AND OTHER APPROPRIATE LARGER (AREA OF
C   RESERVOIR APPROACHING AT LEAST ONE GRID CELL) RESERVOIRS
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lok_ws_to_larger_res(ia, numd, wsuply, sumrain, rain,
     & jday, total_afact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'resws.inc'

      DIMENSION numd(12), rain(max_ncells), sumrain(max_ncells),
     & total_afact(max_ncells), wsuply(max_n_res),
     & adjust_dur_deficit1(max_ncells), adjust_dur_excess0(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (MONTH .EQ. 1) THEN
```

```
         PREV = RSIAPM(12)
      ELSE
         PREV = RSIAPM(MONTH - 1)
      ENDIF

      RSIAP = PREV + IDAY * (RSIAPM(MONTH) - PREV) / NUMD(MONTH)
      IF (DAY1) THEN
         DO i = 1,max_ncells
           adjust_dur_deficit1(i) = 0.0
           adjust_dur_excess0(i) = 0.0
         ENDDO
      ENDIF

         IF ((STAGELO .GE. RSIAP .AND. IDIRFRLO(INDXAWS(IA)) .EQ. 0)
     $   .OR. IDIRFRLO(INDXAWS(IA)) .EQ. 1) THEN
            IB = ibsn_no_res(indxaws(ia))
            DPTHGW = (ELLS(NODEWA(INDXAWS(IA)))-H(NODEWA(INDXAWS(IA))))
     $      * S(NODEWA(INDXAWS(IA)))
            IF (POND(NODEWA(INDXAWS(IA))) .GT. DPTHGW) THEN
               STAGE = POND(NODEWA(INDXAWS(IA))) - DPTHGW + ELLS(NODEWA
     $         (INDXAWS(IA)))
            ELSE
               STAGE = (POND(NODEWA(INDXAWS(IA)))-DPTHGW)/S(NODEWA
     $         (INDXAWS(IA))) + ELLS(NODEWA(INDXAWS(IA)))
            ENDIF
            DPTHSW = AMAX1(STAGE - ELLS(NODEWA(INDXAWS(IA))),0.0)
            IF (STAGE .LE. ELLS(NODEWA(INDXAWS(IA)))) THEN
               AFACT = 1.0
            ELSE
               AFACT = total_afact(NODEWA(INDXAWS(IA)))
            ENDIF
            IF (afact .LE. 1.0) THEN
               sgn = 1
            ELSE
               sgn = -1
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            adjust_for_ovlf = sum_chg_dpth_ovl_flw(NODEWA(INDXAWS(IA)))
            term1_for_adjust_area = (SUMET_outside_res(NODEWA(INDXAWS
     $      (IA))) + seep_loss_res(NODEWA(INDXAWS(IA)))
     $      - SUMRAIN(NODEWA(INDXAWS(IA))) /12.0
     $      - sum_chg_depth_levee_spg(NODEWA(INDXAWS(IA))))
     $      * ((1.0 - afact) / afact)
c
            term2_for_adjust_area = (outf_fc_frm_res_to_cell(nodewa
     $      (indxaws(ia))) +outf_seep_loc_to_res_cell(nodewa(indxaws(ia)
     $      )) +daily_outf_fc_frm_res_to_cell(nodewa(indxaws(ia)))
     $      +adjust_for_ovlf) / afact
c
            IF (sgn * term1_for_adjust_area .LT. term2_for_adjust_area
     $      .AND. DPTHSW .GT. 0.0) THEN
               adjust_dur_excess0(nodewa(indxaws(ia))) = adjust_for_area
     $         (nodewa(indxaws(ia)))+adjust_dur_deficit1(nodewa(indxaws
     $         (ia)))
            ELSE
               adjust_dur_deficit1(nodewa(indxaws(ia))) =
     $         adjust_dur_excess0(nodewa(indxaws(ia)))
```

```
               adjust_dur_excess0(nodewa(indxaws(ia))) =
     $         adjust_dur_deficit1(nodewa(indxaws(ia)))
            ENDIF
            DPTH = DPTHSW / AFACT + term1_for_adjust_area -
     $      adjust_dur_excess0(NODEWA(INDXAWS(IA))) -
     $      (outf_fc_frm_res_to_cell(NODEWA(indxaws(ia))) +
     $      outf_seep_loc_to_res_cell(NODEWA(indxaws(ia))) +
     $      daily_outf_fc_frm_res_to_cell(nodewa(indxaws(ia))) +
     $      sum_chg_dpth_ovl_flw(NODEWA(INDXAWS(IA)))) / afact
            IF (sgn * term1_for_adjust_area .GE. term2_for_adjust_area
     $      .OR. DPTHSW .EQ. 0.0) THEN
               adjust_for_area(NODEWA(INDXAWS(IA))) =
     $         term1_for_adjust_area - term2_for_adjust_area +
     $         daily_excess_adjust(NODEWA(indxaws(ia)))
               IF (afact .LE. 1.0) THEN
                  adjust_for_area(NODEWA(INDXAWS(IA))) = amax1
     $            (adjust_for_area(NODEWA(INDXAWS(IA))),0.0)
               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               adjust_for_area(NODEWA(INDXAWS(IA))) = sgn * amin1(sgn *
     $         adjust_for_area(NODEWA(INDXAWS(IA))),sgn * (DPTH-DPTHSW))

               IF (DPTHSW .EQ. 0.0 .AND. DPTH .GT. 0.000001) THEN
                  adjust_for_area(NODEWA(INDXAWS(IA))) = daily_ET(NODEWA
     $            (INDXAWS(IA))) + daily_seep_loss_res (nodewa(indxaws
     $            (ia))) - chg_depth_levee_spg (NODEWA(INDXAWS(IA))) -
     $            daily_outf_fc_frm_res_to_cell (NODEWA(INDXAWS(IA))) -
     $            daily_outf_seep_loc_to_res_cell (NODEWA(INDXAWS(IA)))
     $            - RAIN(NODEWA(INDXAWS(IA))) / 12.
                  adjust_for_area(NODEWA(INDXAWS(IA))) = amax1
     $            (adjust_for_area(NODEWA(INDXAWS(IA))),0.0)
                  adjust_for_area(NODEWA(INDXAWS(IA))) = amin1
     $            (adjust_for_area(NODEWA(INDXAWS(IA))),DPTH)
               ENDIF
               DPTH = DPTH - adjust_for_area(NODEWA(INDXAWS(IA)))
               stage_res(NODEWA(INDXAWS(IA)),1) = ells(NODEWA
     $         (INDXAWS(IA))) + DPTH
            ENDIF
            IF(DPTH .LT. DPHWS(INDXAWS(IA))) THEN
               DO 10 IY = MAXY,1,-1
                  DO 10 NODE = ISUM(IY) + 1,ISUM(IY + 1)
                     IF (IBSN(NODE) .EQ. IB) THEN
                        DPTHGW = (ELLS(NODE) - H(NODE)) * S(NODE)
                        IF (POND(NODE) .GT. DPTHGW) THEN
                           STAGE = POND(NODE) - DPTHGW + ELLS(NODE)
                        ELSE
                           STAGE = (POND(NODE) - DPTHGW) /S(NODE) + ELLS
     $                     (NODE)
                        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
                        IF (STAGE .GE. ELLS(NODE) .OR.
     $                  sum_chg_dpth_ovl_flw (NODE) .LT. -0.0001) THEN
                           IF (STAGE .LE. ELLS(NODE)) THEN
                              AFACT = 1.0
                           ELSE
                              AFACT = total_afact(NODE)
```

```
                              ENDIF
                              IF (afact .LE. 1.0) THEN
                                 sgn = 1
                              ELSE
                                 sgn = -1
                              ENDIF
                              adjust_for_ovlf = sum_chg_dpth_ovl_flw(NODE)
                              term1_for_adjust_area = (SUMET_outside_res
     $                        (NODE) + seep_loss_res(NODE) - SUMRAIN(NODE)
     $                        /12.0 - sum_chg_depth_levee_spg(NODE))
     $                        * ((1.0 -afact) / afact)
c
                              term2_for_adjust_area =
     $                        (outf_fc_frm_res_to_cell(node) +
     $                        outf_seep_loc_to_res_cell(node) +
     $                        daily_outf_fc_frm_res_to_cell(node) +
     $                        adjust_for_ovlf) / afact
c
                              IF (sgn * term1_for_adjust_area .LT.
     $                        term2_for_adjust_area .AND. STAGE .GT. ELLS
     $                        (NODE)) THEN
                                 adjust_dur_excess0(node) = adjust_for_area
     $                           (node) + adjust_dur_deficit1(node)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                              ELSE
                                 adjust_dur_deficit1(node) =
     $                           adjust_dur_excess0(node)
                                 adjust_dur_excess0(node) =
     $                           adjust_dur_deficit1(node)
                              ENDIF
                              SWDPH = (STAGE-ELLS(NODE)) / AFACT +
     $                        term1_for_adjust_area - adjust_dur_excess0
     $                        (node) - (outf_fc_frm_res_to_cell(NODE) +
     $                        outf_seep_loc_to_res_cell(NODE) +
     $                        daily_outf_fc_frm_res_to_cell(node))/AFACT -
     $                        sum_chg_dpth_ovl_flw(NODE) / AFACT
                              IF (sgn * term1_for_adjust_area .GE.
     $                        term2_for_adjust_area .OR. STAGE .LE. ELLS
     $                        (NODE)) THEN
                                 adjust_for_area(NODE) =
     $                           term1_for_adjust_area -
     $                           term2_for_adjust_area +
     $                           daily_excess_adjust(NODE)
                                 IF (afact .LE. 1.0) THEN
                                    adjust_for_area(NODE) = amax1
     $                              (adjust_for_area(NODE),0.0)
                                 ENDIF
                                 adjust_for_area(NODE) = sgn * amin1(sgn
     $                           * adjust_for_area(NODE) ,sgn * (SWDPH -
     $                           (STAGE - ELLS(NODE))))
                                 IF (STAGE .LE. ELLS(NODE) .AND. SWDPH .GT.
     $                           0.0) THEN
                                    adjust_for_area(NODE) = daily_ET(NODE)
     $                              - rain(NODE)/ 12. + daily_seep_loss_res
     $                              (NODE)-chg_depth_levee_spg(NODE) -
     $                              daily_outf_fc_frm_res_to_cell(NODE) -
     $                              daily_outf_seep_loc_to_res_cell(NODE)
```

```
                                 adjust_for_area(NODE) = amin1
     $                           (adjust_for_area(NODE) ,SWDPH)
                              ENDIF
                              SWDPH = SWDPH - adjust_for_area(NODE)
                           ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                           STAGE_NODE = SWDPH + ELLS(NODE)
                           WSUPLY(INDXAWS(IA)) = WSUPLY(INDXAWS(IA)) +
     $                     (DPHWS(INDXAWS(IA)) - (STAGE_NODE -
     $                     ELLS(NODE))) * GDAR * SFACTOR(INDXAWS(IA))
                           stage_res(node,1) = STAGE_NODE
                        ELSE
                           WSUPLY(INDXAWS(IA))=WSUPLY(INDXAWS(IA))+
     $                     DPHWS(INDXAWS(IA)) *SFACTOR(INDXAWS(IA))
     $                     *GDAR + (ELLS(NODE)-STAGE)*S(NODE)*GDAR
                           stage_res(node,1) = stage
                        ENDIF
                     ENDIF
   10          CONTINUE
               WSUPLY(INDXAWS(IA)) = AMAX1(WSUPLY(INDXAWS(IA)) / 86400.,
     $         0.0)
            ELSE
               WSUPLY(INDXAWS(IA)) = 0.0
            ENDIF
         ELSE
            WSUPLY(INDXAWS(IA)) = 0.0
         ENDIF
         IF (resname(indxaws(ia)) .EQ. name_res_spec_code(3)) THEN
            WSDEL_TO_STA(INDXAWS(IA)) = WSUPLY(INDXAWS(IA))
         ELSEIF (resname(indxaws(ia)) .EQ. name_res_spec_code(19)) THEN
            WSDEL_TO_STA(INDXAWS(IA)) = WSUPLY(INDXAWS(IA))
         ELSE IF(resname(indxaws(ia)) .EQ. name_res_spec_code(4)) THEN
            WSDEL_TO_STA(INDXAWS(IA)) = WSUPLY(INDXAWS(IA))
         ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(6)) THEN
            indx_sta1e_frm_lok_ws = indxaws(ia)
         ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(20)) THEN
            indx_sta2w_frm_lok_ws = indxaws(ia)
         ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(21)) THEN
            indx_sta1ee_frm_lok_ws = indxaws(ia)
         ELSE IF (resname(indxaws(ia)) .EQ. 'STA1W ') THEN
            indx_sta1w_frm_lok_ws = indxaws(ia)
         ENDIF
C
      RETURN
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

lok_ws_to_small_res.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lok_ws_to_small_res.F,v 1.10 2005/06/10 13:29:12 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lok_ws_to_small_res.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------
C   THIS SUBROUTINE COMPUTES WATER SUPPLY RELEASES FROM LAKE OKEECHOBEE
C   TO MAINTAIN SMALLER RESERVOIRS THAT ARE SEPARATE ENTITIES WITHIN MODEL
C   DOMAIN AT DESIRED MINIMUM DEPTHS
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lok_ws_to_small_res (ia, numd, wsuply)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'resbudg.inc'

      DIMENSION numd(12), wsuply(max_n_res)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (MONTH .EQ. 1) THEN
         PREV = RSIAPM(12)
      ELSE
         PREV = RSIAPM(MONTH - 1)
      ENDIF
```

```
      RSIAP = PREV + IDAY * (RSIAPM(MONTH) - PREV) / NUMD(MONTH)

      IF ((STAGELO .GE. RSIAP .AND. IDIRFRLO(INDXAWS(IA)) .EQ. 0)
     $   .OR. IDIRFRLO(INDXAWS(IA)) .EQ. 1) THEN

c  compute depth of water in reservoir

         node = NODEWA(INDXAWS(IA))
         DO ir = 1,noresincell(node)
            IF (ia .eq. ires_index_in_cell(node,ir)) THEN
              depth_in_res = stage_res(node,ir) - ells_in_res(NODE,ir)
            ENDIF
         ENDDO
         IF(depth_in_res .LT. DPHWS(INDXAWS(IA))) THEN
c
c  depth of water in reservoir is less then critical depth; must import
c  water from LOK to keep depth of water in reservoir at critical depth
c
            DO 10 IY = MAXY,1,-1
            DO 10 NODE = ISUM(IY) + 1,ISUM(IY + 1)
               DO k = 1,nnodes(ia)
               IF (NODE .EQ. node_reserv(ia,k)) THEN
                  DO ir = 1,noresincell(node)
                  IF (ia .eq. ires_index_in_cell(node,ir)) THEN
                  IF (stage_res(NODE,ir) .LT. ells_in_res(NODE,ir)) THEN
                     WSUPLY(INDXAWS(IA)) = WSUPLY(INDXAWS(IA)) +
     $               DPHWS(INDXAWS(IA)) *(res_area(INDXAWS(IA))
     $               * 43560./NNODES(INDXAWS(IA))) +
     $               (ells_in_res(NODE,ir)-stage_res(node,ir))*S(NODE)
     $               * GDAR * sfactor(ia)
                  ELSE
                     WSUPLY(INDXAWS(IA)) = WSUPLY(INDXAWS(IA)) +
     $               (DPHWS(INDXAWS(IA))-stage_res(node,ir))*(res_area
     $               (INDXAWS(IA))* 43560./NNODES(INDXAWS(IA)))
                  ENDIF
                  ENDIF
                  ENDDO
               ENDIF
               ENDDO
  10        CONTINUE
            WSUPLY(INDXAWS(IA)) = AMAX1(WSUPLY(INDXAWS(IA))
     $            / 86400.,0.0)
         ELSE
            WSUPLY(INDXAWS(IA)) = 0.0
         ENDIF
      ELSE
         WSUPLY(INDXAWS(IA)) = 0.0
      ENDIF
      IF (resname(indxaws(ia)) .EQ. name_res_spec_code(3)) THEN
         WSDEL_TO_STA(INDXAWS(IA)) = WSUPLY(INDXAWS(IA))
      ELSE IF(resname(indxaws(ia)) .EQ. name_res_spec_code(4)) THEN
         WSDEL_TO_STA(INDXAWS(IA)) = WSUPLY(INDXAWS(IA))
      ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(6)) THEN
         indx_sta1e_frm_lok_ws = indxaws(ia)
      ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(20))THEN
         indx_sta2w_frm_lok_ws = indxaws(ia)
```

```
      ELSE IF (resname(indxaws(ia)) .EQ. name_res_spec_code(21))THEN
         indx_sta1ee_frm_lok_ws = indxaws(ia)
      ELSE IF (resname(indxaws(ia)) .EQ. 'STA1W ') THEN
         indx_sta1w_frm_lok_ws = indxaws(ia)
      ENDIF
C
      RETURN
      END
```

lokres.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: lokres.inc,v 1.3 2003/07/03 20:08:01 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lokres.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /lokres/ s354_res,s351_res,s352_res,dmnds4_res ,
     +dmnds236_res,dmndsemind_res,rlokreswsroten,flpresws_res ,
     +r298dws_res,l8ws_res,wsdelfrwpb_res,capac_struc_to_lokres ,
     +outflow_to_lokres,stagelo_res_beg,fract_rain_litzone
```

loktarg.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: loktarg.inc,v 1.4 2003/07/03 20:08:01 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/loktarg.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /loktarg/ no_targ_loc(max_n_eaa_condt),
     &itarg_area(max_n_targ_loc,max_n_eaa_condt),
     &iserv_area(max_n_eaa_condt),nsacnl(max_n_serv_area),
     &ntotal_no_import_areas,ntotal_no_import_areas_const,
     &lrunoff_to_res_when_above_targ,opt_sim_lok_eaares
C--------------------------
      character lrunoff_to_res_when_above_targ*5,opt_sim_lok_eaares*6
```

lvseep.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: lvseep.F,v 1.21 2005/06/10 13:29:12 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lvseep.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE CALCULATES SEEPAGE BENEATH PROTECTIVE LEVEES
C   INTO BORROW CANALS
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE lvseep (idymax, lvseep_to_l33, lvseep_to_c6, prevqu,
     & rain, crmin, volwtr, kflo0, jday, tseep_divers, tempf_volwtr,
     & tempf_weircff, depth_levee_spg_max_from_gw, sumrain_unadj,
     & rlvseep_frm_cnl, total_small_afact, node_wd, ilvflw_for_slope,
     & HWSEEP_F_L30,HDSEEP_F_L30,ilv_f_l30,no_of_l30spnod,
     & icol_f_l30,irow_f_l30,ipos_lv_l30,ltotal,daily_seep,seepmon)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'lvseepdiv.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'canalloc.inc'
      INCLUDE 'cnlneeds.inc'
      INCLUDE 'dlycslope.inc'
      INCLUDE 'specdssout.inc'
```

```
      DIMENSION crmin(max_ncnls,max_n_seas), daily_seep(max_n_lvsp_loc),
     & depth_levee_spg_max_from_gw(max_ncells),
     & hdiff_res(max_n_res_in_cell,2), hdiffu_res(max_n_res_in_cell,2),
     & head_cellr(2),head1_res(max_n_res_in_cell,2),
     & head3(2), hw_seep_cell(2), iup_cell_no(2),
     & ilvflw_for_slope(max_n_lvsp_loc,2),
     & node_wd(max_ncnls,max_n_weir_outf),
     & prev_rlvseep_to_cnl(max_ncnls), prevqu(max_ncnls),
     & rain(max_ncells), res_length(max_n_res_in_cell,2),
     & rlvseep_frm_cnl(max_ncnls), seepcfs_res(max_n_res_in_cell,2),
     & seepmon(max_n_lvsp_loc), stgdn(2), sumrain_unadj(max_ncells),
     & tempf_volwtr(max_n_res), tempf_weircff(max_n_res,max_nostr_res),
     & total_res_length(2), total_seep_res_cell(2),
     & total_small_afact(max_ncells), tseep_divers(max_no_lvsp),
     & volwtr(max_n_res),ilv_f_l30(10),HWSEEP_F_L30(10),
     & HDSEEP_F_L30(10),icol_f_l30(10),irow_f_l30(10),ipos_lv_l30(10)

      REAL kflo0(max_n_struc_flw,366), lvseep_divers, lvseep_pump_cap,
     & lvseep_to_c6, lvseep_to_l33

      DOUBLE PRECISION crmin,downstrm_canal_stg,CANAL_STAGE_UP,
     & CANAL_STAGE_DN,STGDN,HDIFF,stg_incr_init,rnodal_stage,
     & stgdiff_canal_dn,stg_incr,stage_diff_prev,stage_diff
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      DO 10 N = 1,NCH+1
         DQU(N) = 0.0
         rlvseep_frm_cnl(N) = 0.0
         rlvseep_to_cnl(N) = 0.0
   10 CONTINUE

      IF(IDAY .EQ. 1) THEN
         L=0
         DO 30 IV=1,NLEVS
            DO 20 INODE=1,LVNODES(IV)
               L=L+1
               SEEPMON(L)=0.0
   20       CONTINUE
   30    CONTINUE
         LTOTAL = L
      ENDIF
      ilvcount = 0
      ilvc = 0
      IJK = 0
CLMB<<
      IF (runmode.eq.'CALIB' .and. ifyr+iyear-1 .lt. 1991) then
         l31n_spec_indx = 32
      ELSE
         l31n_spec_indx = 27
      ENDIF
      L = 0
      DO 160 IV = 1,NLEVS
         TSEEPGE(IV) = 0.0
         TSEEP_DIVERS(IV) = 0.0
         DO 150 INODE = 1, LVNODES(IV)
            L = L + 1
```

```
            N3 = LVSP(L,1) - MINX(LVSP(L,2)) + 1 + ISUM(LVSP(L,2))
            N4 = LVSP(L,6) - MINX(LVSP(L,7)) + 1 + ISUM(LVSP(L,7))
            N4_EAST = N4 + 1
            GO TO (40,50,60,70) LVSP(L,3)
   40       N1 = LVSP(L,1) - MINX(LVSP(L,2)) + 1 + ISUM(LVSP(L,2))
            N2 = LVSP(L,5)
            iup_cell_no(1) = n1
            iup_cell_no(2) = n4
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (NODCR(N2) .GT. 1) THEN
              STGDIFF_CANAL_DN = FLOAT(NODCR(N2)-ICNL_NODE_POS(L,2))/
     $        FLOAT(NODCR(N2)-1) * HDC(N2,iseas)
            ELSE
              STGDIFF_CANAL_DN = 0.0
            ENDIF
            CANAL_STAGE_DN = CHDEP(N2) + STGDIFF_CANAL_DN
C
            IF (LVSP(L,1) .EQ. 26 .AND. LVSP(L,2) .EQ. 22 .AND. CNM(N2)
     $      .EQ. canl_names_spec(l31n_spec_indx) .AND. IBSN(N4) .EQ. 13
     $      .AND. IBSN(N4_EAST) .EQ. 0) THEN
               STGDN(1) = CHDEP(IC4) + HDC(IC4,iseas)
            ELSEIF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 27) THEN
               STGDN(1) = (CHDEP(ISNCRE) + HDC(ISNCRE,iseas)
     $          + CHDEP(N2) + HDC(N2,iseas)) / 2.0
            ELSE
               STGDN(1) = CANAL_STAGE_DN
            ENDIF
C
            DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
            IF (POND(N1) .GT. DPTHGW) THEN
               HW_SEEP_orig = POND(N1) - DPTHGW + ELLS(N1)
            ELSE
               HW_SEEP_orig = (POND(N1)-DPTHGW)/S(N1) + ELLS(N1)
            ENDIF
            HW_SEEP = HW_SEEP_orig
            STGDN(2) = HW_SEEP
c
c  for reservoir cell calculate seepage at maximum depth
c     for determination of avail storage
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            DPTHGW = (ELLS(N4) - H(N4)) * S(N4)
            IF (POND(N4) .GT. DPTHGW) THEN
               HEAD3(1) = POND(N4) - DPTHGW + ELLS(N4)
            ELSE
               HEAD3(1) = (POND(N4)-DPTHGW)/S(N4) + ELLS(N4)
            ENDIF
            HEAD3(2) = HW_SEEP
            DO kk = 1,2
               NCELLR = iup_cell_no(kk)
               DPTHGW = (ELLS(NCELLR) - H(NCELLR)) * S(NCELLR)
               IF (POND(NCELLR) .GT. DPTHGW) THEN
                  HW_SEEP_cell(kk) = POND(NCELLR) - DPTHGW
     $            + ELLS(NCELLR)
               ELSE
                  HW_SEEP_cell(kk) = (POND(NCELLR)-DPTHGW)/S(NCELLR)
     $            + ELLS(NCELLR)
```

```
                ENDIF
                DO ir = 1,noresincell(NCELLR)
                   RES_LENGTH(ir,kk) = 0.0
                   ia = ires_index_in_cell(NCELLR,ir)
                   IF ((sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $             (ia) .eq. 'NO ') .AND. resname(ia) .NE. 'L37L33') THEN
                      IF (HW_SEEP_cell(kk) .LT. ELLS(NCELLR)) stage_res
     $                (NCELLR,ir)=HW_SEEP_cell(kk)
                      HW_SEEP_outside_res = HW_SEEP_cell(kk)
                      IF (HW_SEEP_cell(kk) .GT. ELLS(NCELLR)) THEN
                        depth_of_deficit_c =SUMET_outside_res(ncellr)
     $                   +seep_loss_res_cell(ncellr)- SUMRAIN_unadj
     $                   (ncellr)/12.0-sum_chg_depth_levee_spg_unadj
     $                   (ncellr)-(outf_fc_frm_res_to_cell(ncellr) +
     $                   sum_chg_dpth_ovl_flw(ncellr))
                         HW_SEEP_outside_res = ells(ncellr) - amin1
     $                   (depth_of_deficit_c ,0.0)
                      ENDIF
                      IF (HW_SEEP_cell(kk) .GT. ELLS(NCELLR)) THEN
                         IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $                   THEN
                            HW_SEEP_res = stage_res(ncellr,ir)
                            HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                            HEAD1_res(ir,kk) = HW_SEEP_res
                            HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)
     $                       - HEAD3(kk)
                            RES_LENGTH(ir,kk) = sqrt(res_area(ia)/640.)
                         ENDIF
                      ENDIF
                   ELSE
                      DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
                      IF (POND(N1) .GT. DPTHGW) THEN
                         HW_SEEP = POND(N1) - DPTHGW
     $                   + ELLS(N1)
                      ELSE
                         HW_SEEP = (POND(N1)-DPTHGW)
     $                   /S(N1) + ELLS(N1)
                      ENDIF
                      HW_SEEP_outside_res = HW_SEEP
                      IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $                THEN
                         HW_SEEP_res = stage_res(ncellr,ir)
                         HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                         HEAD1_res(ir,kk) = HW_SEEP_res
                         HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)-HEAD3(kk)
c                        print *,HW_SEEP_res,STGDN(kk),HEAD3(kk)
c    $                   ,HDIFF_res(ir,kk),HDIFFU_res(ir,kk),'  LPLPLP'
                         IF (width_of_res(ia) .GT. 0.0) THEN
                           IF (nnodes(ia) .eq. 1) THEN
                              RES_LENGTH(ir,kk) = (res_area(ia)/640.)
     $                        /width_of_res(ia)
                           ELSE
                             if (ncellr .eq. node_reserv(ia,nnodes(ia)))
     $                       THEN
                                RES_LENGHR = (res_area(ia)/640.)
     $                          /width_of_res(ia)
                                RES_LENGTH(ir,kk) =RES_LENGHR-(nnodes(ia)
```

```
     $                        - 1)*DY/5280.
                           ELSE
                              RES_LENGTH(ir,kk) = DY/5280.
                           ENDIF
                         ENDIF
                       ELSE
                          RES_LENGTH(ir,kk) = sqrt(sfactor(ia)*GDAR/
     $                    (5280.*5280))
                       ENDIF
                     ENDIF
                  ENDIF
c              print *,HW_SEEP_outside_res,depth_of_deficit_c,'  XXXX'
                  HW_SEEP = HW_SEEP_outside_res
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                 HDIFF = HW_SEEP - STGDN(1)
                 HEAD1 = HW_SEEP
                 HDIFFU = HEAD1 - HEAD3(1)
                 IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES'
     $           .AND. (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $           (ia) .eq. 'NO ')) THEN
                    IF (HW_SEEP_orig .le.  ELLS(N1)) THEN
                      HW_SEEP_res = HW_SEEP_orig
                      HDIFF_res(ir,kk) = HDIFF
                      HEAD1_res(ir,kk) = HEAD1
                      HDIFFU_res(ir,kk) = HDIFFU
                    ENDIF
                 ENDIF
              END DO
           END DO
           HDIFF = HW_SEEP - STGDN(1)
           HEAD1 = HW_SEEP
           HDIFFU = HEAD1 - HEAD3(1)
           if (CNM(N2) .EQ. 'L30  ') THEN
              IJK = IJK + 1
              HWSEEP_F_L30(IJK) = HW_SEEP
              HDSEEP_F_L30(IJK) = HEAD3(1)
              ilv_f_l30(IJK) = IV
              no_of_l30spnod = IJK
              icol_f_l30(IJK) = LVSP(L,1)
              irow_f_l30(IJK) = LVSP(L,2)
              ipos_lv_l30(IJK) = L
           endif
           GO TO 80
   50      N1 = LVSP(L,4)
           N2 = LVSP(L,6) - MINX(LVSP(L,7)) + 1 + ISUM(LVSP(L,7))
c
           iup_cell_no(1) = n2
           iup_cell_no(2) = n3
           IF (NODCR(N1) .GT. 1) THEN
             STGDIFF_CANAL_UP = FLOAT(NODCR(N1) - ICNL_NODE_POS(L,1)) /
     $       FLOAT(NODCR(N1)-1) * HDC(N1,iseas)
           ELSE
             STGDIFF_CANAL_UP = 0.0
           ENDIF
           CANAL_STAGE_UP = CHDEP(N1) + STGDIFF_CANAL_UP
           STGDN(1) = CANAL_STAGE_UP
c
```

```
            DPTHGW = (ELLS(N3) - H(N3)) * S(N3)
            IF (POND(N3) .GT. DPTHGW) THEN
               HEAD1_1 = POND(N3) - DPTHGW + ELLS(N3)
            ELSE
               HEAD1_1 = (POND(N3)-DPTHGW)/S(N3) + ELLS(N3)
            ENDIF
            HEAD_CELLR(1) = HEAD1_1
            DPTHGW = (ELLS(N2) - H(N2)) * S(N2)
            IF (POND(N2) .GT. DPTHGW) THEN
               HEAD3_1 = POND(N2) - DPTHGW + ELLS(N2)
            ELSE
               HEAD3_1 = (POND(N2)-DPTHGW)/S(N2) + ELLS(N2)
            ENDIF
            STGDN(2) = HEAD3_1
            HEAD_CELLR(2) = HEAD3_1
c
c  for reservoir cell calculate seepage at maximum depth
c     for determination of avail storage
c
            HDIFF = CANAL_STAGE_UP - HEAD3_1
            HDIFFU = HEAD1_1 - HEAD3_1
            HW_SEEP = CANAL_STAGE_UP
            DO kk = 1,2
               ncellr = iup_cell_no(kk)
               DPTHGW = (ELLS(NCELLR) - H(NCELLR)) * S(NCELLR)
               IF (POND(NCELLR) .GT. DPTHGW) THEN
                  HW_SEEP_cell(kk) = POND(NCELLR) - DPTHGW
     $            + ELLS(NCELLR)
               ELSE
                  HW_SEEP_cell(kk) = (POND(NCELLR)-DPTHGW)/S(NCELLR)
     $            + ELLS(NCELLR)
               ENDIF
               DO ir = 1,noresincell(ncellr)
                  RES_LENGTH(ir,kk) = 0.0
                  ia = ires_index_in_cell(ncellr,ir)
                  IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $            (ia) .eq. 'NO ') THEN
                     HW_SEEP_outside_res = CANAL_STAGE_UP
                     IF (HW_SEEP_cell(kk) .LT. ELLS(ncellr)) stage_res
     $                (ncellr,ir) = HW_SEEP_cell(kk)
                     IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $               THEN
                        HW_SEEP_res = stage_res(ncellr,ir)
                        HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                        HEAD1_res(ir,kk) = HW_SEEP_res
                        HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)
     $                  - HEAD_CELLR(kk)
                        RES_LENGTH(ir,kk) = sqrt(res_area(ia)/640.)
                     ENDIF
                  ELSE
                     HW_SEEP = CANAL_STAGE_UP
                     HW_SEEP_outside_res = HW_SEEP
                     IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $               THEN
                        HW_SEEP_res = stage_res(ncellr,ir)
                        HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                        HEAD1_res(ir,kk) = HW_SEEP_res
```

```
                        HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)
     $                  - HEAD_CELLR(kk)
                        IF (width_of_res(ia) .GT. 0.0) THEN
                          IF (nnodes(ia) .eq. 1) THEN
                             RES_LENGTH(ir,kk) = (res_area(ia)/640.)
     $                       /width_of_res(ia)
                          ELSE
                             if (ncellr .eq. node_reserv(ia,nnodes(ia)))
     $                       THEN
                                RES_LENGHR = (res_area(ia)/640.)
     $                          /width_of_res(ia)
                                RES_LENGTH(ir,kk)=RES_LENGHR-(nnodes(ia)
     $                          - 1)*DY/5280.
                             ELSE
                                RES_LENGTH(ir,kk) = DY/5280.
                             ENDIF
                          ENDIF
                        ELSE
                          RES_LENGTH(ir,kk)=sqrt(sfactor(ia)*GDAR/(5280.
     $                    *5280))
                        ENDIF
                     ENDIF
                  ENDIF
c              print *,HW_SEEP_outside_res,depth_of_deficit_c,'  XXXX'
                  HW_SEEP = HW_SEEP_outside_res
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                 HDIFF = HW_SEEP - HEAD3_1
                 HDIFFU = HEAD1_1 - HEAD3_1
               END DO
            END DO
            HDIFF = HW_SEEP - HEAD3_1
            HDIFFU = HEAD1_1 - HEAD3_1
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            GO TO 80
   60       N1 = LVSP(L,4)
            N2 = LVSP(L,5)
            iup_cell_no(1) = N3
            iup_cell_no(2) = N4
            IF (NODCR(N2) .GT. 1) THEN
              STGDIFF_CANAL_DN = FLOAT(NODCR(N2)-ICNL_NODE_POS(L,2))/
     $        FLOAT(NODCR(N2)-1) * HDC(N2,iseas)
            ELSE
              STGDIFF_CANAL_DN = 0.0
            ENDIF
            CANAL_STAGE_DN = CHDEP(N2) + STGDIFF_CANAL_DN
            STGDIFF_CANAL_UP = FLOAT(NODCR(N1)-ICNL_NODE_POS(L,1))/
     $      FLOAT(NODCR(N1))* HDC(N1,iseas)
            CANAL_STAGE_UP = CHDEP(N1) + STGDIFF_CANAL_UP
            HDIFF = CANAL_STAGE_UP - CANAL_STAGE_DN
            GO TO 80
   70       N1 = LVSP(L,1) - MINX(LVSP(L,2)) + 1 + ISUM(LVSP(L,2))
            N2 = LVSP(L,6) - MINX(LVSP(L,7)) + 1 + ISUM(LVSP(L,7))
            iup_cell_no(1) = n1
            iup_cell_no(2) = n2
            DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
            IF (POND(N1) .GT. DPTHGW) THEN
               STGUP = POND(N1) - DPTHGW + ELLS(N1)
```

```
            ELSE
               STGUP = (POND(N1)-DPTHGW)/S(N1) + ELLS(N1)
            ENDIF
            STGDN(2) = STGUP
            DPTHGW = (ELLS(N2) - H(N2)) * S(N2)
            IF (POND(N2) .GT. DPTHGW) THEN
               STGDN(1) = POND(N2) - DPTHGW + ELLS(N2)
            ELSE
               STGDN(1) = (POND(N2)-DPTHGW)/S(N2) + ELLS(N2)
            ENDIF
c
c  for reservoir cell calculate seepage at maximum depth
c     for determination of avail storage
c
            HDIFF = STGUP - STGDN(1)
            HDIFFU = HDIFF
            DO kk = 1,2
               NCELLR = iup_cell_no(kk)
               DPTHGW = (ELLS(NCELLR) - H(NCELLR)) * S(NCELLR)
               IF (POND(NCELLR) .GT. DPTHGW) THEN
                  HW_SEEP_cell(kk) = POND(NCELLR) - DPTHGW
     $            + ELLS(NCELLR)
               ELSE
                  HW_SEEP_cell(kk) = (POND(NCELLR)-DPTHGW)/S(NCELLR)
     $            + ELLS(NCELLR)
               ENDIF
               HW_SEEP = HW_SEEP_cell(1)
               DO ir = 1,noresincell(NCELLR)
                  RES_LENGTH(ir,kk) = 0.0
                  ia = ires_index_in_cell(NCELLR,ir)
                  IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $            (ia) .eq. 'NO ') THEN
                     IF (HW_SEEP_cell(kk) .LT. ELLS(NCELLR)) stage_res
     $               (NCELLR,ir)=HW_SEEP_cell(kk)
                     HW_SEEP_outside_res = HW_SEEP_cell(kk)
                     IF (HW_SEEP_cell(1) .GT. ELLS(N1)) THEN
                       depth_of_deficit_c =SUMET_outside_res(n1)
     $                  +seep_loss_res_cell(n1)- SUMRAIN_unadj
     $                  (n1)/12.0-sum_chg_depth_levee_spg_unadj
     $                  (n1)-(outf_fc_frm_res_to_cell(n1) +
     $                  sum_chg_dpth_ovl_flw(n1))
                        HW_SEEP_outside_res = ells(n1) - amin1
     $                  (depth_of_deficit_c ,0.0)
                     ENDIF
                     IF (HW_SEEP_cell(kk) .GT. ELLS(NCELLR)) THEN
                        IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $                  THEN
                           HW_SEEP_res = stage_res(ncellr,ir)
                           HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                           HEAD1_res(ir,kk) = HW_SEEP_res
                           HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)
     $                      - STGDN(kk)
                           RES_LENGTH(ir,kk) = sqrt(res_area(ia)/640.)
                        ENDIF
                     ENDIF
                  ELSE
                     DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
```

```
                     IF (POND(N1) .GT. DPTHGW) THEN
                        HW_SEEP = POND(N1) - DPTHGW
     $                  + ELLS(N1)
                     ELSE
                        HW_SEEP = (POND(N1)-DPTHGW)
     $                  /S(N1) + ELLS(N1)
                     ENDIF
                     HW_SEEP_outside_res = HW_SEEP
                     IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $               THEN
                        HW_SEEP_res = stage_res(ncellr,ir)
                        HDIFF_res(ir,kk) = HW_SEEP_res - STGDN(kk)
                        HEAD1_res(ir,kk) = HW_SEEP_res
                        HDIFFU_res(ir,kk) = HEAD1_res(ir,kk)-STGDN(kk)
                        IF (width_of_res(ia) .GT. 0.0) THEN
                          IF (nnodes(ia) .eq. 1) THEN
                             RES_LENGTH(ir,kk) = (res_area(ia)/640.)
     $                       /width_of_res(ia)
                          ELSE
                            if (ncellr .eq. node_reserv(ia,nnodes(ia)))
     $                      THEN
                               RES_LENGHR = (res_area(ia)/640.)
     $                         /width_of_res(ia)
                               RES_LENGTH(ir,kk) =RES_LENGHR-(nnodes(ia)
     $                         - 1)*DY/5280.
                            ELSE
                               RES_LENGTH(ir,kk) = DY/5280.
                            ENDIF
                          ENDIF
                        ELSE
                           RES_LENGTH(ir,kk) = sqrt(sfactor(ia)*GDAR/
     $                     (5280.*5280))
                        ENDIF
                     ENDIF
                  ENDIF
c              print *,HW_SEEP_outside_res,depth_of_deficit_c,'  XXXX'
                  HW_SEEP = HW_SEEP_outside_res
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                 HDIFF = HW_SEEP - STGDN(1)
                 HEAD1 = HW_SEEP
                 HDIFFU = HEAD1 - STGDN(1)
                 IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES'
     $           .AND. (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $           (ia) .eq. 'NO ')) THEN
                    IF (HW_SEEP_orig .le.  ELLS(N1)) THEN
                      HW_SEEP_res = HW_SEEP_orig
                      HDIFF_res(ir,kk) = HDIFF
                      HEAD1_res(ir,kk) = HEAD1
                      HDIFFU_res(ir,kk) = HDIFFU
                    ENDIF
                 ENDIF
              END DO
           END DO
   80         IF (SRATE(IV,1) .LT. 0.0) THEN
                 HD_DROP = MAX(HDIFF,0.0)
              ELSE
                 HD_DROP = HDIFF
```

```
              ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
            IF (SRATE(IV,2) .LT. 0.0) THEN
               HD_DROP_N = AMAX1(HDIFFU,0.0)
            ELSE
               HD_DROP_N = HDIFFU
            ENDIF
            SEEPCFS = SRATE(IV,1)*HD_DROP+SRATE(IV,2)*HD_DROP_N + SRATE
     $      (IV,3)
            DO kk = 1,2
               DO ir = 1,noresincell(iup_cell_no(kk))
                  ia = ires_index_in_cell(iup_cell_no(kk),ir)
                  IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $            THEN
                     IF (SRATE(IV,2) .LT. 0.0) THEN
                        HD_DROP_N_res = amax1(HDIFFU_res(ir,kk),0.0)
                     ELSE
                        HD_DROP_N_res = HDIFFU_res(ir,kk)
                     ENDIF
                     IF (SRATE(IV,1) .LT. 0.0) THEN
                        HD_DROP_res = amax1(HDIFF_res(ir,kk),0.0)
                     ELSE
                        HD_DROP_res = HDIFF_res(ir,kk)
                     ENDIF
                     seepcfs_res(ir,kk)=SRATE(IV,1)*HD_DROP_res+SRATE
     $               (IV,2)*HD_DROP_N_res + SRATE(IV,3)
c                    print *,seepcfs_res(ir,kk),SRATE(IV,1),HD_DROP_res
c    $               ,SRATE(IV,2),HD_DROP_N_res,SRATE(IV,3)
c    $               ,resname(ia),'  LVSEEPR'
                  ELSE
                     seepcfs_res(ir,kk) = 0.0
                  ENDIF
               ENDDO
            ENDDO
            IF(LVNAME(IV) .EQ. 'L40E ') THEN
               IF (SEEPCFS .LT. -0.66) SEEPCFS = -0.66
            ELSEIF (LVNAME(IV) .NE. 'L38E ' .AND. LVNAME(IV) .NE.
     $       'L-8 ' .AND. LVNAME(IV) .NE. 'ACMEB' ) THEN
               IF (SEEPCFS .LT. 0.0) SEEPCFS = 0.0
            ENDIF
            SEEPCFS0 = SEEPCFS * srate_frac(IV)
            SEEPCFS0 = amin1(SEEPCFS0,rate_limit(IV))
            SEEPCFS = SEEPCFS0 * DISTNC(LVSP(L,8))/5280.
            DO kk = 1,2
               TOTAL_RES_LENGTH(kk) = 0.0
               total_add_cell_lvseep = 0.0
               DO ir = 1,noresincell(iup_cell_no(kk))
                 ia = ires_index_in_cell(iup_cell_no(kk),ir)
                 IF (ires_lev_seep_dir_cnl_opt(ia) .EQ. 'YES')
     $           THEN
                   TOTAL_RES_LENGTH(kk) = TOTAL_RES_LENGTH(kk) +
     $             RES_LENGTH(ir,kk)
                   total_add_cell_lvseep = total_add_cell_lvseep
     $             + SEEPCFS0*RES_LENGTH(ir,kk)*(1.0 -
     $             frac_seep_dir_cnl(ia))
                   IF (ir .eq. noresincell(iup_cell_no(kk))) THEN
```

```
                    SEEPCFS1 = SEEPCFS0 * amax1(DISTNC(LVSP(L,8))/5280.
     $              - TOTAL_RES_LENGTH(kk),0.0) + total_add_cell_lvseep
                    seepcfs = SEEPCFS1
                  ENDIF
                  seepcfs_res(ir,kk) = SEEPCFS_res(ir,kk) *
     $            RES_LENGTH(ir,kk)* frac_seep_dir_cnl(ia)
c          print *,TOTAL_RES_LENGTH(kk),seepcfs_res(ir,kk),SEEPCFS1
c    $     ,seepcfs,'  YYY  ',resname(ia)
                ENDIF
               END DO
            END DO
C
            GO TO (90,110,120,130) LVSP(L,3)
c
c    make sure seepage rate does not result in reverse of
c     gradient
c
   90       IF (lvname(iv) .NE. 'L-8 '.and. cnm(n2).ne.'L30  ') THEN
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
c     iterate to find rate of seepage that would result in
c     stage to west of levee and canal stage in borrow canal
c     being equal. This rate would be the upper limit for
c     time step.
c
c     initially set canal stage equal to nodal stage to west
c     of levee
               DPTHGW = (ELLS(N1) - H(N1)) * S(N1)
               IF (POND(N1) .GT. DPTHGW) THEN
                  rnodal_stage = POND(N1) - DPTHGW + ELLS(N1)
               ELSE
                  rnodal_stage = (POND(N1)-DPTHGW)/S(N1) + ELLS(N1)
               ENDIF
               stgdiff_canal_dn = 0.0
               do kk = 1,nodcr(n2)
                 if (LVSP(L,6) .eq. xcn(n2,kk) .and. LVSP(L,7)
     $            .eq. ycn(n2,kk)) then
                   if (nodcr(n2) .gt. 1) then
                     stgdiff_canal_dn = (float(nodcr(n2)-kk)/float(nodcr
     $               (n2)-1))*hdc(n2,iseas)
                   else
                     stgdiff_canal_dn = 0.0
                   endif
                   go to 95
                 endif
               enddo
  95           canal_stage_dn = chdep(n2)+ stgdiff_canal_dn
c     perform iteration
c              if (cnm(n2).eq.'L33 ') print *,rnodal_stage
c    $             ,canal_stage_dn,chdep(n2),' LLLL'
               IF (rnodal_stage .GE. canal_stage_dn) THEN
                  stg_incr_init = max((rnodal_stage-canal_stage_dn)
     $            /5.0,0.001)
                  stg_incr = stg_incr_init
                  iter_s = 1
                  stage_diff_prev = 1.
  100             coutlim = 0.0
```

```
                    downstrm_canal_stg = rnodal_stage-stgdiff_canal_dn
                    tot_pump = 0.0
                    DO is = 1,nstrctr(n2)
                       coutlim1 = strout(downstrm_canal_stg ,n2,
     $                 n2,is,0,crmin,volwtr,prevqu,rain,0,tot_pump,jday ,
     $                 tempf_volwtr,tempf_weircff,0)
                       coutlim = coutlim + coutlim1
c                      if (cnm(n2) .eq. 'L30  ') print *,tdqu_dn(idncnl(n2
c    $                 ,is)),downstrm_canal_stg,coutlim,flnm(istr(n2,is))
c    $                 ,'  KKK  ',cnm(idncnl(n2,is)),upstream_inflow_lim
                    END DO
                    DO is = 1,nout(n2)
                       IF (into(n2,is) .GE. 0) THEN
                          IF (into(n2,is) .NE. 999) THEN
                            IF (downstrm_canal_stg .GT. crel(n2,is)) THEN
                              coutlim = coutlim+strcf(n2,is)*gwdth(n2,is)
     $                        *(downstrm_canal_stg-crel(n2,is)) *sqrt
     $                        (downstrm_canal_stg-crel(n2,is))
                            ENDIF
                          ELSE
                            lstnode = NODE_WD(n2,is)
                            hw = downstrm_canal_stg
                            deptgw = ells(lstnode) - h(lstnode)
                            crelev = crel(n2,is)
                            if (pond(lstnode) .gt. deptgw) then
                              tw = pond(lstnode) - deptgw + ells(lstnode)
                            else
                              tw = (pond(lstnode) - deptgw)/s(lstnode)
     &                        + ells(lstnode)
                            endif
                            IF (tw .le. crelev) THEN
                               IF (hw.gt.crelev) THEN
                                 vol_lim_w = (hw - ells(lstnode))* gdar
     &                            * 0.80 / 86400.
                                 cout = strcf(n2,is)*gwdth(n2,is)*(hw
     &                           -crelev)*sqrt(hw-crelev)
                                 cout = amin1(cout,vol_lim_w)
                               ELSE
                                 cout = 0.0
                               ENDIF
                            ELSE
                               IF (hw .lt. crelev) THEN
                                  sgn = -1.0
                                  vol_lim_w = amax1(pond(lstnode)
     &                            -deten(lstnode),0.0)*gdar/86400.
                                  cout = sgn*strcf(n2,is)*gwdth(n2
     &                            ,is)*(tw-crelev)*sqrt(tw-crelev)
                                  cout = sgn*amin1(sgn*cout,vol_lim_w)
                               ELSE
                                  IF (hw .gt. tw) THEN
                                    submerg_fact = (tw - crelev)
     &                              /(hw - crelev)
                                    rel_crst_length = (hw - crelev)
     &                              /(hw - crelev + 5.0)
                                    rmod_limit = 0.85 - 0.50*
     &                              rel_crst_length
                                    reduc_fact = (submerg_fact-rmod_limit)
```

```
     &                            /(1.0 -rmod_limit)
                                  reduc_fact = amax1(reduc_fact,0.0)
                                  reduc_fact = (1.0-reduc_fact) ** 0.25
                                  vol_lim_w = (hw - tw) * gdar * 0.80
     &                            /86400.
                                  cout = strcf(n2,is)*gwdth(n2,is)*
     &                            (tw-crelev)*sqrt(tw-crelev)*reduc_fact
                                  cout = amin1(cout,vol_lim_w)
                                ELSE
                                  sgn = -1.0
                                  submerg_fact = (hw - crelev)
     &                            /(tw - crelev)
                                  rel_crst_length = (tw - crelev)
     &                            /(tw - crelev + 5.0)
                                  rmod_limit = 0.85 - 0.50*
     &                            rel_crst_length
                                  reduc_fact = (submerg_fact-rmod_limit)
     &                            /(1.0 -rmod_limit)
                                  reduc_fact = amax1(reduc_fact,0.0)
                                  reduc_fact = (1.0-reduc_fact) ** 0.25
                                  cout = sgn*strcf(n2,is)*gwdth(n2
     &                            ,is)*(tw-crelev)*sqrt(tw-crelev)
     &                            *reduc_fact
                                  vol_lim_w = amax1(pond(lstnode)
     &                            -deten(lstnode),0.0)*gdar*0.9/86400.
                                  cout = sgn*amin1(sgn*cout,vol_lim_w)
                                ENDIF
                              ENDIF
                            ENDIF
                            coutlim = coutlim + cout
                          ENDIF
                        ENDIF
                      ENDDO
c
                      canal_stage_dn_end = canal_stage_dn - stgdiff_canal_dn
                      rmax_vol_canal =(downstrm_canal_stg
     $                -canal_stage_dn_end)*cl(n2) * width(n2) - cgseep(n2,
     $                downstrm_canal_stg,0,0.)-tovflc(n2 ,
     $                downstrm_canal_stg,0,0.,0,1) + coutlim * 86400. -
     $                prevqu(n2) + prev_rlvseep_to_cnl(n2)*86400. -
     $                rlvseep_to_cnl(n2)*86400.
                      surf_water_dpth_incr = amin1(rmax_vol_canal/(gdar*
     $                (1.0-total_small_afact(n1))),pond(n1))
                      surf_water_dpth = pond(n1) - surf_water_dpth_incr
                      gw_level = h(n1) - amax1(rmax_vol_canal/(gdar*(1.0-
     $                total_small_afact(n1)))-surf_water_dpth_incr,0.0)
     $                /s(n1)
                      stage_node = surf_water_dpth + gw_level
                      stage_diff = stage_node - rnodal_stage
c                     if (cnm(n2) .eq. 'L33  ') print *,coutlim,prevqu(n2)
c    $                 /86400.,'  SEEPLIM',ITER_S,rmax_vol_canal/86400.
c    $                 ,canal_stage_dn_end,downstrm_canal_stg
c    $                 ,stage_node,rnodal_stage,chdep(n2),stgdiff_canal_dn
c    $                 ,prev_rlvseep_to_cnl(n2),rlvseep_to_cnl(n2)
c    $                 ,month,iday
                      IF (stage_diff .NE. 0.0) THEN
                         IF (stage_diff/stage_diff_prev .LT. 0.0 .AND.
```

```
     $               iter_s .NE. 1) THEN
                        stg_incr = - abs(stg_incr) * 0.5 * stage_diff
     $                  / abs(stage_diff)
                     ENDIF
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
                  IF (iter_s .EQ. 1 .AND. stage_diff .GT. 0.0)stg_incr =
     $            -stg_incr_init
                  IF (abs(stage_diff) .LT. 0.05 .OR. iter_s .GT. 100)
     $            THEN
                     frac_of_limit = 0.75
                     seeplim = rmax_vol_canal / 86400.*frac_of_limit
c                    if (iv .eq. 2) print *,seepcfs,seeplim
c    $               ,'  LIMCOMP'
                     seepcfs = amin1(seepcfs,seeplim)
                     seepcfs = amax1(seepcfs,0.0)
                  ELSE
                     iter_s = iter_s + 1
                     stage_diff_prev = stage_diff
                     rnodal_stage = rnodal_stage - stg_incr
                     go to 100
                  ENDIF
               ELSE
                  seeplim = 0.0
                  seepcfs = 0.0
               ENDIF
            ENDIF
            IF (CNM(N2) .EQ. 'L30   ') THEN
               seepcfs = 0.0
               seeplim = 0.0
            ENDIF
c
            IF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 29) THEN
               ilvc = ilvc + 1
               IF (runmode .EQ. 'CALIB') THEN
                  IF (ilvc .GT. 1) THEN
                     dpth_seep = amin1(SEEPCFS,kflo0(ifls31,jday))
     $               * 86400. / (GDAR*(1.0-total_small_afact(n1)
     $               - carfact(n1)))
                  ELSE
                     DPTH_SEEP = SEEPCFS * 86400. / (GDAR*(1.0
     $               - total_small_afact(n1)-carfact(n1)))
                  ENDIF
               ELSE
                  DPTH_SEEP = SEEPCFS * 86400. / (GDAR*(1.0
     $               - total_small_afact(n1)-carfact(n1)))
               ENDIF
            ELSE
               DPTH_SEEP = SEEPCFS * 86400. / (GDAR*(1.0
     $               - total_small_afact(n1)-carfact(n1)))
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
            IF (SEEPCFS .gt. 0.000001) THEN
               SEEP_GW = AMAX1(DPTH_SEEP - POND(N1),0.0)
               chg_depth_levee_spg(n1) = - amin1(DPTH_SEEP,POND(n1))
            ELSE
               sgn = -1.0
```

```
                gw_stor_cell = (ELLS(N1)-H(N1))*S(N1)
                chg_depth_levee_spg(n1) = amax1(sgn*DPTH_SEEP
     $          - gw_stor_cell,0.0)
             ENDIF
             DO kk = 1,2
             total_seep_res_cell(kk) = 0.0
             NCELLR = iup_cell_no(kk)
             DO ir = 1,noresincell(ncellr)
                ia = ires_index_in_cell(ncellr,ir)
                IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     $          (ia) .eq. 'NO ') THEN
                  volwtr(ia) = volwtr(ia) - chg_depth_levee_spg(ncellr)
     $            * GDAR/ 86400. * sfactor(ia)
                ELSE
                  DPTH_SEEP_RES = seepcfs_res(ir,kk)*1.9835/(sfactor(ia)
     $            * gdar/43560.)
c              print *,resname(ia),pond_in_res(ncellr,ir),'  IN LVSEEP'
c    $         ,DPTH_SEEP_RES,ells_in_res(ncellr,ir)
                  IF (seepcfs_res(ir,kk) .gt. 0.000001) THEN
                     sgn = 1.0
                     if (frac_seep_dir_cnl(ia) .lt. 1.0) then
                        rmax_lseep_res =amax1(stage_res(ncellr,ir)-amax1(
     $                  ells_in_res(ncellr,ir),h(ncellr)+pond(ncellr))
     $                  ,0.0) + amax1(amin1(ells_in_res(ncellr,ir),
     $                  stage_res(ncellr,ir)) - (h(ncellr)+pond(ncellr))
     $                  ,0.0) * s(ncellr)
                     else
                        rmax_lseep_res = 99999.
                     endif
                     DPTH_SEEP_RES = amin1(DPTH_SEEP_RES
     $               ,rmax_lseep_res)
                     seep_gw_res = amax1(DPTH_SEEP_RES-pond_in_res(ncellr
     $               ,ir),0.0)
                     stage_res(ncellr,ir) = stage_res(ncellr,ir) -
     $               amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
     $               - seep_gw_res/s(ncellr)
                     pond_in_res(ncellr,ir) = pond_in_res(ncellr,ir) -
     $               amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
                  ELSE
                     sgn = -1.0
                     seepcfs_res(ir,kk) = 0.0
                     DPTH_SEEP_RES = abs(seepcfs_res(ir,kk))*1.9835
     $               /(sfactor(ia)*gdar/43560.)
                     gw_stor_res = amax1(ells_in_res(ncellr,ir)-stage_res
     $                (ncellr,ir),0.0)* s(ncellr)
                     stage_res(ncellr,ir) = stage_res(ncellr,ir)
     $               + amin1(gw_stor_res,DPTH_SEEP_RES)/s(ncellr)
     $               + amax1(DPTH_SEEP_RES-gw_stor_res,0.0)
                     pond_in_res(ncellr,ir) = amax1(stage_res(ncellr,ir)
     $               - ells_in_res(ncellr,ir),0.0)
                  ENDIF
                  volwtr(ia) = volwtr(ia) + sgn*DPTH_SEEP_RES
     $            *sfactor(ia)*gdar/86400.
                  res_lvseep_out(ia) = res_lvseep_out(ia)
     $              + sgn*DPTH_SEEP_RES*sfactor(ia)*gdar/43560.
                  seepcfs_res(ir,kk) = sgn*DPTH_SEEP_RES*gdar*sfactor(ia)
     $            / 86400.
```

```
c                 print *,resname(ia),pond_in_res(ncellr,ir)
c     $           ,' IN LVSEEP2  ',stage_res(ncellr,ir)
                  total_seep_res_cell(kk) = total_seep_res_cell(kk)
      $           + seepcfs_res(ir,kk)
               ENDIF
            END DO
            END DO
            IF (seepcfs .gt. 0.000001) THEN
               IF (H(n1)+POND(n1) .GT. ELLS(n1)) THEN
                  sum_chg_depth_levee_spg_unadj(n1) =
      $           sum_chg_depth_levee_spg_unadj(n1) - DPTH_SEEP
                  sum_chg_depth_levee_spg(n1) = sum_chg_depth_levee_spg
      $           (n1)- amin1(DPTH_SEEP,POND(n1))
               ENDIF
               POND(N1) = POND(N1) - AMIN1(DPTH_SEEP,POND(N1))
               RCHG(N1) = RCHG(N1) - SEEP_GW
               depth_levee_spg_max_from_gw(n1) = seep_gw
            ELSE
               sgn = -1.0
               gw_stor_cell = (ELLS(n1)-H(n1))*S(n1)
               IF (H(n1)+POND(n1) .GT. ELLS(n1) .OR.
      $        sgn*DPTH_SEEP .GT. gw_stor_cell) THEN
                  sum_chg_depth_levee_spg(n1) = sum_chg_depth_levee_spg
      $           (n1) + chg_depth_levee_spg(n1)
                  depth_levee_spg_max_from_gw(n1) = - amin1(gw_stor_cell
      $           ,sgn*DPTH_SEEP)
               ENDIF
               RCHG(N1) = RCHG(N1) + amin1(gw_stor_cell,sgn*DPTH_SEEP)
               POND(N1) = POND(n1) + amax1(sgn*DPTH_SEEP-gw_stor_cell
      $         ,0.0)
            ENDIF
            total_seep_cell = SEEPCFS + total_seep_res_cell(1)
C
c
            IF (nint(total_seep_cell) .GE. 0) THEN
            IF (runmode .EQ. 'SIMUL') THEN
               lvseep_divers(l) = amin1(total_seep_cell
      $        ,lvseep_pump_cap(l)) *frac_seep_divers(l,iseas)
            ELSE
               lvseep_divers(l) = 0.0
            ENDIF
            IF (LVSP(L,1) .EQ. 26 .AND. LVSP(L,2) .EQ. 22 .AND. CNM(N2)
      $     .EQ. canl_names_spec(l31n_spec_indx) .AND. IBSN(N4) .EQ. 13
      $     .AND. IBSN(N4_EAST) .EQ. 0) THEN
               IF (cnm(IC4) .EQ. canl_names_spec(8)) THEN
                  IF (chdep(ic4) .GE. 6.5) THEN
                     frac_c4 = 0.10
                     frac_l31n = 0.90
                  ELSEIF (chdep(ic4) .GE. 2.5) THEN
                     frac_c4 = 0.10 + ((6.5- chdep(ic4))**3.0/
      $              ((6.5-2.5) ** 3.0)) * 0.10
                     frac_l31n = 0.90 - ((6.5 - chdep(ic4))**3.0/
      $              ((6.5-2.5) ** 3.0)) * 0.10
                  ELSE
                     frac_c4 = 0.20
                     frac_l31n = 0.80
                  ENDIF
```

```
                 QU (IC4) = QU (IC4) + total_seep_cell
     $            * 86400. * frac_c4
                 QU (N2) = QU (N2) + total_seep_cell*86400.
     $            * frac_l31n
                 qu(ic4) = qu(ic4) - lvseep_divers(l) * 86400.* frac_c4
                 qu(n2) = qu(n2) - lvseep_divers(l) * 86400.* frac_l31n
                 rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $           + (total_seep_cell - lvseep_divers(l))*frac_l31n
                 IF (ivary_slope_daily_opt(IC4) .eq. 1) THEN
                   call match(cnm(IC4),1,5,canal_name_dvslope
     &             ,ncnl_dly_slope,5,imatch_slope)
                   dischg_for_slope_kn(imatch_slope,1,1)
     &             = dischg_for_slope_kn(imatch_slope,1,1)
     &             + (total_seep_cell-lvseep_divers(l)) * frac_c4
                 ENDIF
                 IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                   call match(cnm(N2),1,5,canal_name_dvslope
     &             ,ncnl_dly_slope,5,imatch_slope)
                   dischg_for_slope_kn(imatch_slope,1,2)
     &             = dischg_for_slope_kn(imatch_slope,1,2)
     &             + (total_seep_cell-lvseep_divers(l)) * frac_l31n
                 ENDIF
              ELSE
                 QU (IC4) = QU (IC4) + total_seep_cell * 86400. * 0.20
                 QU (N2) = QU (N2) + total_seep_cell * 86400. * 0.80
                 qu(ic4) = qu(ic4) - lvseep_divers(l) * 86400. * 0.20
                 qu(n2) = qu(n2) - lvseep_divers(l) * 86400. * 0.80
                 rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $           + (total_seep_cell - lvseep_divers(l)) * 0.80
                 IF (ivary_slope_daily_opt(IC4) .eq. 1) THEN
                   call match(cnm(IC4),1,5,canal_name_dvslope
     &             ,ncnl_dly_slope,5,imatch_slope)
                   dischg_for_slope_kn(imatch_slope,1,1)
     &             = dischg_for_slope_kn(imatch_slope,1,1)
     &             + (total_seep_cell-lvseep_divers(l)) * 0.20
                 ENDIF
                 IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                   call match(cnm(N2),1,5,canal_name_dvslope
     &             ,ncnl_dly_slope,5,imatch_slope)
                   dischg_for_slope_kn(imatch_slope,1,2)
     &             = dischg_for_slope_kn(imatch_slope,1,2)
     &             + (total_seep_cell-lvseep_divers(l)) * 0.80
                 ENDIF
              ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
           ELSEIF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 27) THEN
              NODE2826 =  28 - MINX(26) + 1 + ISUM(26)
              IF(ELLS(NODE2826) .GT. 0.0) THEN
                 QU (ISNCRE) = QU (ISNCRE) + total_seep_cell * 86400.
     $           * 0.60
                 QU (L30) = QU(L30) + total_seep_cell * 86400. * 0.40
                 QU (ISNCRE) = QU (ISNCRE) - lvseep_divers(l) * 86400.
     $           *0.60
                 QU (L30) = QU(L30) - lvseep_divers(l) * 86400. * 0.40
                 rlvseep_to_cnl(L30) = rlvseep_to_cnl(L30)
     $               + (total_seep_cell - lvseep_divers(l)) * 0.40
                 IF (ivary_slope_daily_opt(L30) .eq. 1) THEN
```

```
                 call match(cnm(L30),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
                 dischg_for_slope_kn(imatch_slope,1,2)
     &           = dischg_for_slope_kn(imatch_slope,1,2)
     &           + (total_seep_cell-lvseep_divers(l)) * 0.40
              ENDIF
              IF (ivary_slope_daily_opt(ISNCRE) .eq. 1) THEN
                 call match(cnm(ISNCRE),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
                 dischg_for_slope_kn(imatch_slope,1,1)
     &           = dischg_for_slope_kn(imatch_slope,1,1)
     &           + (total_seep_cell-lvseep_divers(l)) * 0.60
              ENDIF
           ELSE
              QU (L30) = QU(L30) + total_seep_cell * 86400.
              QU (L30) = QU(L30) - lvseep_divers(l) * 86400.
              rlvseep_to_cnl(L30) = rlvseep_to_cnl(L30)
     $        + (total_seep_cell - lvseep_divers(l))
              IF (ivary_slope_daily_opt(L30) .eq. 1) THEN
                 call match(cnm(L30),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
                 dischg_for_slope_kn(imatch_slope,1,2)
     &           = dischg_for_slope_kn(imatch_slope,1,2)
     &           + (total_seep_cell-lvseep_divers(l))
              ENDIF
           ENDIF
        ELSEIF (LVSP(L,1) .EQ. 27 .AND. LVSP(L,2) .EQ. 29) THEN

           ilvcount = ilvcount + 1
c
           IF (ilvcount .EQ. 1) THEN
              lvseep_to_l33 = seepcfs
              QU(N2) = QU(N2) + lvseep_to_l33 * 86400.
              QU(N2) = QU(N2) - lvseep_divers(l) * 86400.
              rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $        + (lvseep_to_l33-lvseep_divers(l))
              IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                 call match(cnm(N2),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
                 dischg_for_slope_kn(imatch_slope,1,3)
     &           = dischg_for_slope_kn(imatch_slope,1,3)
     &           - (lvseep_to_l33-lvseep_divers(l))
              ENDIF
           ELSE
              IF (runmode .EQ. 'CALIB') THEN
                 seep_to_c6 = amin1(seepcfs,kflo0(ifls31,jday))
              ELSE
                 seep_to_c6 = seepcfs
              ENDIF
              lvseep_to_c6 = seep_to_c6
              QU(N2) = QU(N2) + SEEP_TO_C6 * 86400.
              QU(N2) = QU(N2) - lvseep_divers(l) * 86400.
              rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $        + (lvseep_to_c6-lvseep_divers(l))
              IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                 call match(cnm(N2),1,5,canal_name_dvslope
     &           ,ncnl_dly_slope,5,imatch_slope)
```

```
                  dischg_for_slope_kn(imatch_slope,1,1)
     &              = dischg_for_slope_kn(imatch_slope,1,1)
     &              + (lvseep_to_c6-lvseep_divers(l))
                 ENDIF
               ENDIF
c
            ELSE
               QU (N2) = QU (N2) + total_seep_cell * 86400.
               rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $         + (total_seep_cell-lvseep_divers(l))
               IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                  call match(cnm(N2),1,5,canal_name_dvslope
     &            ,ncnl_dly_slope,5,imatch_slope)
                  sgn = 1
                  if (ilvflw_for_slope(l,2) .eq. 3) sgn = -1
                  dischg_for_slope_kn(imatch_slope,1,
     $            ilvflw_for_slope(l,2)) = dischg_for_slope_kn
     $            (imatch_slope,1,ilvflw_for_slope(l,2))
     $            + sgn *(total_seep_cell-lvseep_divers(l))
               ENDIF
               QU(N2) = QU(N2) - lvseep_divers(l) * 86400.
            ENDIF
            ELSE
              lvseep_divers(l) = 0.0
              QU (N2) = QU (N2) + total_seep_cell * 86400.
              rlvseep_to_cnl(n2) = rlvseep_to_cnl(n2)
     $         + (total_seep_cell-lvseep_divers(l))
              IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
                 call match(cnm(N2),1,5,canal_name_dvslope
     $           ,ncnl_dly_slope,5,imatch_slope)
                 sgn = 1
                 if (ilvflw_for_slope(l,2) .eq. 3) sgn = -1
                 dischg_for_slope_kn(imatch_slope,1,
     $           ilvflw_for_slope(l,2)) = dischg_for_slope_kn
     $           (imatch_slope,1,ilvflw_for_slope(l,2))
     $           + sgn *(total_seep_cell-lvseep_divers(l))
              ENDIF
              QU(N2) = QU(N2) - lvseep_divers(l) * 86400.
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
C
            GO TO 140
c
c  calculate volume limit for seepage during time step
c
  110       stgdiff_canal_up = 0.0
            do kk = 1,nodcr(n1)
              if (LVSP(L,1) .eq. xcn(n1,kk) .and. LVSP(L,2)
     $         .eq. ycn(n1,kk)) then
               if (nodcr(n1) .gt. 1) then
                 stgdiff_canal_up = (float(nodcr(n1)-kk)/float(nodcr
     $           (n1)-1))*hdc(n1,iseas)
               else
                 stgdiff_canal_up = 0.0
               endif
               go to 115
```

```
              endif
            enddo
  115       canal_stage_up = chdep(n1)+ stgdiff_canal_up
            if (cnm(n1) .ne. 'CA1 ') then
               depth_limit = max(canal_stage_up,ells(n2)) - max
     $         (ells(n2),h(n2)+pond(n2))+(min(canal_stage_up,ells(n2))
     $         - min(ells(n2),h(n2)+pond(n2)))*s(n2)
                depth_limit = amax1(depth_limit,0.0)
                seep_limit = depth_limit * gdar/86400. * 0.75
            else
                seep_limit = 9999.
            endif
            seepcfs = amin1(SEEPCFS,seep_limit)
c
            QU (N1) = QU (N1) - SEEPCFS * 86400.
            IF (ivary_slope_daily_opt(N1) .eq. 1) THEN
               call match(cnm(N1),1,5,canal_name_dvslope
     $         ,ncnl_dly_slope,5,imatch_slope)
               sgn = 1
               if (ilvflw_for_slope(l,1) .eq. 2) sgn = -1
               dischg_for_slope_kn(imatch_slope,1,
     $         ilvflw_for_slope(l,1)) = dischg_for_slope_kn
     $         (imatch_slope,1,ilvflw_for_slope(l,1))
     $         + sgn *SEEPCFS
            ENDIF
            depth_lvseep = SEEPCFS*86400./(GDAR*(1.0
     $      - total_small_afact(n2)-carfact(n2)))
c
            IF (SEEPCFS .gt. 0.000001) THEN
               sgn = 1.0
               POND(N2) = POND(N2) + depth_lvseep

               IF (runmode .EQ. 'SIMUL') THEN
                  lvseep_divers(l) = sgn*amin1(sgn*seepcfs
     $            ,lvseep_pump_cap(l)) *frac_seep_divers(l,iseas)
               ELSE
                  lvseep_divers(l) = 0.0
               ENDIF
            ELSE
               sgn = -1.0
               RCHG(N2) = RCHG(N2) - amax1(sgn*depth_lvseep
     $         - POND(N2),0.0)
               POND(N2) = POND(N2) - amin1(sgn*depth_lvseep
     $         ,POND(N2))
               lvseep_divers(l) = 0.0
            ENDIF
c
            total_seep_cell = SEEPCFS
            IF (nint(SEEPCFS) .ge. 0) THEN
               POND(n2) = POND(n2)-lvseep_divers(l)*86400./(GDAR
     $         *(1.0-total_small_afact(n2)-carfact(n2)))
               IF (H(N2)+POND(n2) .GT. ELLS(n2)) THEN
                 sum_chg_depth_levee_spg(n2) =
     $           sum_chg_depth_levee_spg(n2) +
     $           (SEEPCFS-lvseep_divers(l)) * 86400. / GDAR
               ENDIF
               chg_depth_levee_spg(n2) = (SEEPCFS - lvseep_divers(l))
```

```
$         * 86400. /GDAR
       ELSE
          QU(n1) = QU(n1) - sgn*lvseep_divers(l) * 86400.
          IF (H(N2)+POND(n2) .GT. ELLS(n2)) THEN
            sum_chg_depth_levee_spg(n2)=sum_chg_depth_levee_spg(n2)
$           + SEEPCFS * 86400. / GDAR
          ENDIF
          chg_depth_levee_spg(n2) = SEEPCFS *86400./GDAR
       ENDIF
       DO ir = 1,noresincell(n2)
          ia = ires_index_in_cell(n2,ir)
          IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
$         (ia) .eq. 'NO ') THEN
             IF (nint(SEEPCFS) .gt. 0) THEN
                volwtr(ia)=volwtr(ia)-(SEEPCFS-lvseep_divers(l))
$               * sfactor(ia)
             ELSE
                volwtr(ia)=volwtr(ia)-SEEPCFS* sfactor(ia)
             ENDIF
          ELSE
            kk = 1
            total_seep_res_cell(kk) = 0.0
            NCELLR = N2
            ia = ires_index_in_cell(ncellr,ir)
            DPTH_SEEP_RES = seepcfs_res(ir,kk)*1.9835/(sfactor(ia)
$           * gdar/43560.)
            IF (seepcfs_res(ir,kk) .gt. 0.000001) THEN
               sgn = 1.0
               rmax_lseep_res = amax1(stage_res(ncellr,ir)-amax1(
$              ells_in_res(ncellr,ir),h(ncellr)+pond(ncellr)),0.0)
$              + amax1(amin1(ells_in_res(ncellr,ir),stage_res
$              (ncellr,ir)) - (h(ncellr)+pond(ncellr)),0.0)
$              * s(ncellr)
               DPTH_SEEP_RES = amin1(DPTH_SEEP_RES
$              ,rmax_lseep_res)
               seep_gw_res = amax1(DPTH_SEEP_RES-pond_in_res(ncellr
$              ,ir),0.0)
               stage_res(ncellr,ir) = stage_res(ncellr,ir) -
$              amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
$              - seep_gw_res/s(ncellr)
               pond_in_res(ncellr,ir) = pond_in_res(ncellr,ir) -
$              amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
            ELSE
               sgn = -1.0
               seepcfs_res(ir,kk) = 0.0
               DPTH_SEEP_RES = abs(seepcfs_res(ir,kk))*1.9835
$              /(sfactor(ia)* gdar/43560.)
               gw_stor_res = amax1(ells_in_res(ncellr,ir)-stage_res
$               (ncellr,ir),0.0)* s(ncellr)
               stage_res(ncellr,ir) = stage_res(ncellr,ir)
$              + amin1(gw_stor_res,DPTH_SEEP_RES)/s(ncellr)
$              + amax1(DPTH_SEEP_RES-gw_stor_res,0.0)
               pond_in_res(ncellr,ir) = amax1(stage_res(ncellr,ir)
$              - ells_in_res(ncellr,ir),0.0)
            ENDIF
            volwtr(ia) = volwtr(ia) + sgn*DPTH_SEEP_RES
$           *gdar*sfactor(ia)/86400.
```

```
                 res_lvseep_out(ia) = res_lvseep_out(ia)
     $             + sgn*DPTH_SEEP_RES*sfactor(ia)*gdar/43560.
                 seepcfs_res(ir,kk) = sgn*DPTH_SEEP_RES*gdar*sfactor(ia)
     $           / 86400.
c                print *,resname(ia),pond_in_res(ncellr,ir)
c    $           ,' IN LVSEEP2  ',stage_res(ncellr,ir)
                 total_seep_res_cell(kk) = total_seep_res_cell(kk)
     $           + seepcfs_res(ir,kk)
               ENDIF
             END DO
             rlvseep_frm_cnl(n1) = SEEPCFS + rlvseep_frm_cnl(n1)
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
             GO TO 140
  120        QU(N1) = QU(N1) - SEEPCFS * 86400.
             QU(N2) = QU(N2) + SEEPCFS * 86400.
             total_seep_cell = SEEPCFS
             IF (nint(SEEPCFS) .ge. 0) THEN
               IF (runmode .EQ. 'SIMUL') THEN
                 lvseep_divers(l) = amin1(seepcfs
     $           ,lvseep_pump_cap(l)) *frac_seep_divers(l,iseas)
               ELSE
                  lvseep_divers(l) = 0.0
               ENDIF
               qu(n2) = qu(n2) - lvseep_divers(l) * 86400.
             ELSE
               lvseep_divers(l) = 0.0
             ENDIF
             rlvseep_to_cnl(n2) = (SEEPCFS - lvseep_divers(l))
     $       + rlvseep_to_cnl(n2)
             rlvseep_frm_cnl(n1) = SEEPCFS + rlvseep_frm_cnl(n1)
             IF (ivary_slope_daily_opt(N1) .eq. 1) THEN
               call match(cnm(N1),1,5,canal_name_dvslope
     $         ,ncnl_dly_slope,5,imatch_slope)
               sgn = 1
               if (ilvflw_for_slope(l,1) .eq. 2) sgn = -1
               dischg_for_slope_kn(imatch_slope,1,
     $         ilvflw_for_slope(l,1)) = dischg_for_slope_kn
     $         (imatch_slope,1,ilvflw_for_slope(l,1))
     $         + sgn *SEEPCFS
             ENDIF
             IF (ivary_slope_daily_opt(N2) .eq. 1) THEN
               call match(cnm(N2),1,5,canal_name_dvslope
     $         ,ncnl_dly_slope,5,imatch_slope)
               sgn = 1
               if (ilvflw_for_slope(l,2) .eq. 3) sgn = -1
               dischg_for_slope_kn(imatch_slope,1,
     $         ilvflw_for_slope(l,2)) = dischg_for_slope_kn
     $         (imatch_slope,1,ilvflw_for_slope(l,2))
     $         + sgn *(SEEPCFS-lvseep_divers(l))
             ENDIF
C
             GO TO 140
  130        DO kk = 1,2
               total_seep_res_cell(kk) = 0.0
               noder = iup_cell_no(kk)
               DO ir = 1,noresincell(noder)
```

```
                ia = ires_index_in_cell(noder,ir)
                IF (sfactor(ia) .LT. sfactmin .AND. ires_small_sim(ia)
$               .EQ. 'YES') THEN
                   IF (seepcfs_res(ir,kk) .gt. 0.000001) THEN
                      sgn = 1.0
                      DPTH_SEEP_RES = seepcfs_res(ir,kk)*1.9835
$                     /(gdar*sfactor(ia)/43560.)
                      stage_lim = amax1(h(n1)+pond(n1),h(n2)+pond(n2))
                      rmax_lseep_res = amax1(stage_res(ncellr,ir)-
$                     amax1(ells_in_res(ncellr,ir),stage_lim),0.0)
$                     + amax1(amin1(ells_in_res(ncellr,ir),stage_res
$                     (ncellr,ir)) - stage_lim,0.0) * s(ncellr)
                      rmax_lseep_res = rmax_lseep_res*0.75
                      DPTH_SEEP_RES = amin1(DPTH_SEEP_RES
$                     ,rmax_lseep_res)
                      seep_gw_res = amax1(DPTH_SEEP_RES-pond_in_res
$                     (ncellr,ir),0.0)
                      stage_res(ncellr,ir) = stage_res(ncellr,ir) -
$                     amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
$                     - seep_gw_res/s(ncellr)
                      pond_in_res(ncellr,ir) = pond_in_res(ncellr,ir)-
$                     amin1(DPTH_SEEP_RES,pond_in_res(ncellr,ir))
                      seepcfs_res(ir,kk) = DPTH_SEEP_RES*gdar*
$                     sfactor(ia)/86400.
                   ELSE
                      sgn = -1.0
                      DPTH_SEEP_RES = abs(seepcfs_res(ir,kk))*1.9835
$                     /(gdar*sfactor(ia)/43560.)
                      stage_lim = stage_res(noder,ir)
                      if (kk .eq. 1) then
                         stage_cell_lim = h(n2)+pond(n2)
                         depth_cell_lim = pond(n2)
                      else
                         stage_cell_lim = h(n1)+pond(n1)
                         depth_cell_lim = pond(n1)
                      endif
                      if (depth_cell_lim .lt. 0.00001) THEN
                        rmax_lseep_res = 0.0
                      else
                        dpth_limit_pot = depth_cell_lim / sfactor(ia)
                        depth_to_limit_in_res =amax1(amin1(ells_in_res
$                       (noder,ir),stage_lim) - stage_lim,0.0)*
$                       s(noder)+amax1(amin1(stage_cell_lim,stage_lim)
$                       - amax1(stage_lim,ells_in_res(noder,ir)),0.0)
                      endif
                      rmax_lseep_res = rmax_lseep_res*0.75
                      DPTH_SEEP_RES = amin1(DPTH_SEEP_RES
$                     ,rmax_lseep_res)
                      gw_stor_res = amax1(ells_in_res(ncellr,ir) -
$                     stage_res(ncellr,ir),0.0)* s(ncellr)
                      stage_res(ncellr,ir) = stage_res(ncellr,ir)
$                     + amin1(gw_stor_res,DPTH_SEEP_RES)/s(ncellr)
$                     + amax1(DPTH_SEEP_RES-gw_stor_res,0.0)
                      pond_in_res(ncellr,ir) = amax1(stage_res(ncellr
$                     ,ir) - ells_in_res(ncellr,ir),0.0)
                      seepcfs_res(ir,kk) = sgn*DPTH_SEEP_RES*gdar*
$                     sfactor(ia)/86400.
```

```
                  ENDIF
                  volwtr(ia) = volwtr(ia) + sgn*DPTH_SEEP_RES
     $            *gdar*sfactor(ia)/86400.
                  res_lvseep_out(ia) = res_lvseep_out(ia)
     $            + sgn*DPTH_SEEP_RES*sfactor(ia)*gdar/43560.
                  total_seep_res_cell(kk) = total_seep_res_cell(kk)
     $            +  seepcfs_res(ir,kk)
               ENDIF
            END DO
         END DO
         IF (runmode .EQ. 'SIMUL') THEN
            lvseep_divers(l) = amin1(seepcfs+total_seep_res_cell(1)
     $       +total_seep_res_cell(2),lvseep_pump_cap(l))
     $       *frac_seep_divers(l,iseas)
         ELSE
            lvseep_divers(l) = 0.0
         ENDIF
         total_seep_cell = SEEPCFS + total_seep_res_cell(1) +
     $   total_seep_res_cell(2)
         IF (nint(total_seep_cell) .ge. 0) THEN
            RCHG(n1) = RCHG(n1) - amax1(total_seep_cell * 86400.
     $      / (GDAR*(1.0-total_small_afact(n1)-carfact(n1)))
     $      - POND(n1),0.0)
            POND(N1) = POND(N1) - amin1(total_seep_cell* 86400./(GDAR
     $      *(1.0-total_small_afact(n1)-carfact(n1))) ,POND(n1))
            POND(N2) = POND(N2) + total_seep_cell *86400. / (GDAR
     $      *(1.0-total_small_afact(n2)-carfact(n2)))
            POND(n2) = POND(n2) - lvseep_divers(l)*86400./ (GDAR
     $      *(1.0-total_small_afact(n2)-carfact(n2)))
         ELSE
            sgn = -1.0
            RCHG(N2) = RCHG(N2) - amax1(sgn*total_seep_cell*86400.
     $      / (GDAR*(1.0-total_small_afact(n2)-carfact(n2)))
     $      - POND(N2),0.0)
            POND(N2) = POND(N2) - amin1(sgn*total_seep_cell*86400.
     $      / (GDAR*(1.0-total_small_afact(n2)-carfact(n2)))
     $      ,POND(N2))
            POND(N1) = POND(N1) + sgn*total_seep_cell*86400./(GDAR
     $      *(1.0-total_small_afact(n1)-carfact(n1)))
         ENDIF
c
         IF (h(n2)+pond(n2) .GT. ells(n2)) THEN
            sum_chg_depth_levee_spg(n2) = sum_chg_depth_levee_spg(n2)
     $      + (SEEPCFS-lvseep_divers(l)) * 86400. / GDAR
         ENDIF
         chg_depth_levee_spg(n2) = (SEEPCFS - lvseep_divers(l)) *
     $   86400. / GDAR
         IF (h(n1)+pond(n1) .GT. ells(n1)) THEN
            sum_chg_depth_levee_spg(n1) = sum_chg_depth_levee_spg(n1)
     $      - SEEPCFS * 86400. / GDAR
         ENDIF
         chg_depth_levee_spg(n1) = - SEEPCFS * 86400. / GDAR
         DO kk = 1,2
            noder = iup_cell_no(kk)
            DO ir = 1,noresincell(noder)
               ia = ires_index_in_cell(noder,ir)
               IF (sfactor(ia) .GE. sfactmin .OR. ires_small_sim(ia)
```

```
     $            .EQ. 'NO ') THEN
                     volwtr(ia) = volwtr(ia) -chg_depth_levee_spg(noder)
     $                * GDAR/ 86400. * sfactor(ia)
                  ENDIF
               ENDDO
            ENDDO
C
  140       TSEEPGE(IV) = TSEEPGE(IV) + total_seep_cell
            TSEEP_DIVERS(IV) = TSEEP_DIVERS(IV) + lvseep_divers(l)
            IF (opt_dest_lvseep(l) .EQ. 1) THEN
               DO nd = 1,no_dest_lvseep(l)
                  pond(idest_cell_loc_lvseep(l,nd)) = pond
     $            (idest_cell_loc_lvseep(l,nd)) + lvseep_divers(l)
     $            * 86400. / (no_dest_lvseep(l) * gdar * (1.0 -
     $            total_small_afact(idest_cell_loc_lvseep(l,nd))))
               END DO
            ELSE
               DO nd = 1, no_dest_lvseep(l)
                  qu(idest_canal_lvseep_index(l,nd)) = qu
     $            (idest_canal_lvseep_index(l,nd)) + lvseep_divers(l)
     $            * 86400. / no_dest_lvseep(l)
               END DO
            ENDIF
            nodellll = 27 - minx(25) + 1 + isum(25)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C
c daily_seep in acre-ft
c
            daily_seep(L) = total_seep_cell * 1.9835
C SEEPMON IN ACRE-FT
            IF (.NOT. DAY1) THEN
               SEEPMON(L)=SEEPMON(L)+total_seep_cell * 86400. / 43560.
            ENDIF
C
  150    CONTINUE
  160 CONTINUE
C
C
      do n = 1,nch
         prev_rlvseep_to_cnl(n) = rlvseep_to_cnl(n)
      enddo
C
      RETURN
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

lvseepdiv.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: lvseepdiv.inc,v 1.4 2003/07/03 20:08:02 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/lvseepdiv.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /lvseepdiv/ lvseep_divers(max_n_lvsp_loc),
     &lvseep_pump_cap(max_n_lvsp_loc),opt_dest_lvseep(max_n_lvsp_loc),
     &idest_cell_loc_lvseep(max_n_lvsp_loc,max_n_dest_lvsp),
     &idest_canal_lvseep_index(max_n_lvsp_loc,max_n_dest_lvsp),
     &no_dest_lvseep(max_n_lvsp_loc),
     &lvseep_divers_str_name(max_n_lvsp_loc),
     &frac_seep_divers(max_n_lvsp_loc,max_n_seas)
C-----------------------------

      INTEGER opt_dest_lvseep
      CHARACTER lvseep_divers_str_name*6
```

main.F

```
C-----------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: main.F,v 1.44 2005/08/08 23:55:40 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/main.F,v $
C   $Name: rel-5-5 $
C
C
C                SOUTH FLORIDA WATER MANAGEMENT MODEL
C
C                MAJOR VARIABLES ; MAIN PROGRAM
C
C       MAXY            NUMBER OF GRID ROWS IN THE NORTH/SOUTH DIRECTION
C       MINX,MAXX       X GRID LOCATION OF THE WEST AND EAST
C                         BOUNDARIES, RESPECTIVELY, FOR EACH Y
C       DX,DY           GRID SPACING IN X & Y DIRECTIONS
C       ELLS            LAND ELEVATION (FT ABOVE MSL)
C       PUMP            DAILY WELLFIELD PUMPAGE VALUES (MGD)
C       RCHG            GROUNDWATER RECHARGE (FT)
C       POND            SURFACE PONDING (FT)
C       LOCW            WELLFIELD LOCATION SUBSCRIPT
C       NUMD            NUMBER OF DAYS IN EACH MONTH
C       LP              LOGICAL ARRAY OF PRINT SELECTIONS
C       RAIN            LOCAL VARIABLE FOR BASIN RAINFALL
C       IDAY            DAY OF MONTH
C       MONTH           MONTH OF YEAR
C       IYEAR           INDICATOR OF YEAR
C
C-----------------------------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      PROGRAM sfwmm

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'budg.inc'
```

```
 INCLUDE 'c1.inc'
 INCLUDE 'c2.inc'
 INCLUDE 'et.inc'
 INCLUDE 'routc.inc'
 INCLUDE 'stat.inc'
 INCLUDE 'wrout.inc'
 INCLUDE 'stas.inc'
 INCLUDE 'simflows.inc'
 INCLUDE 'wcaot.inc'
 INCLUDE 'lakwca.inc'
 INCLUDE 'cstlucie.inc'
 INCLUDE 'ccaloos.inc'
 INCLUDE 'environ.inc'
 INCLUDE 'resadj.inc'
 INCLUDE 'trig_lec_et_param.inc'
 INCLUDE 'trigger.inc'
 INCLUDE 'lec_et.inc'
 INCLUDE 'asr_param.inc'
 INCLUDE 'asr1.inc'
 INCLUDE 'asr2.inc'
 INCLUDE 'asr3.inc'
 INCLUDE 'addlok.inc'
 INCLUDE 'ovfllim.inc'
 INCLUDE 'resws.inc'
 INCLUDE 'opersched.inc'
 INCLUDE 'STRCOUNT.inc'
 INCLUDE 'wcasim.inc'
 INCLUDE 'flw298dso.inc'
 INCLUDE 'loktarg.inc'
 INCLUDE 'add_misc.inc'
 INCLUDE 'reuse.inc'
 INCLUDE 'lokres.inc'
 INCLUDE 'climvar.inc'
 INCLUDE 'specdssout.inc'
 INCLUDE 'resbudg.inc'
 INCLUDE 'smallres.inc'
 INCLUDE 'stormd.inc'
 INCLUDE 'dualops.inc'
 INCLUDE 'dlycslope.inc'

 DIMENSION ccdmdro(2), dmdronm(10), cfcurfut(2),
& clim_threshold_est(2), offset_to_sched(max_n_stor_w_oper_sch),
& ro_factor_lok(12), rf_factor(12), dpthgw_for_rain(max_ncells),
& ro_factor(12), day_pet0_lok(31), grid_area_res(max_n_res),
& runoff_south_bef_divers(max_n_eaa_condt),
& icol_stg(max_n_targ_area,max_n_targ_loc),
& irow_stg(max_n_targ_area,max_n_targ_loc),
& base_area_eaa_basin(max_n_eaa_condt), fracdph_min(12,3),
& fracdph_max(12,3), rlok_stg_beg_forpmp(max_n_eaa_condt),
& rlok_stg_end_forpmp(max_n_eaa_condt),rmax_dschg_coef(max_ncnls),
& sum_eaa_dmnd_ssm(max_n_eaa_condt), rlvseep_frm_cnl(max_ncnls),
& pot_del_to_eaa_basin(max_n_eaa_condt,2), alocnnr_potdel(2),
& day_rain_nsm(max_n_cells_nsm,31), pond_end(max_ncells),
& rf_factor_lok(12), aqperm_in_res(max_ncells,max_n_res_in_cell),
& total_afact(max_ncells), iunit_no_res_file(max_n_sm_res),
& aqdep_in_res(max_ncells,max_n_res_in_cell),
& deltastor(max_n_res), crmin_bel_flr(max_ncnls,max_n_seas),
```

```
& node_wd(max_ncnls,max_n_weir_outf), tvolwtrl(max_n_res),
& mon_targ_wet(max_n_goal_ssm), iday_targ_wet(max_n_goal_ssm),
& ndsglim(max_ncnls,max_nostr_cnl),ipos_cell_in_canl(max_ncells,3),
& nodedlim(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
& fl_trig_constrt(max_ncnls,max_nostr_cnl,max_n_ds_cell_lim),
& tvolwtrl_p(max_n_res), nrgtrg(max_n_res,max_nostr_res),
& noder_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
& stgr_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
& n_gated_grav_outlets_fc(max_ncnls),n_pump_outlets_fc(max_ncnls)

 DIMENSION iopt_lim_des_cap_wca(max_n_ostruc_wca,max_n_wcas),
& icnl_no_reg_seas(max_n_cnl_w_weir_seas),
& iweir_no_reg_seas(max_n_cnl_w_weir_seas),
& weir_reg_seas(max_n_cnl_w_weir_seas,max_n_weir_outf,max_n_seas),
& avvolgr_mx(max_n_res,max_nostr_res),no_canl_in_cell(max_ncells),
& rmean_cnl_bot_elev(max_n_cnls_w_var_slpe),
& flow_resistvty_coef(max_n_cnls_w_var_slpe),
& iwuppflw_for_slope(max_ncnls,max_n_weir_outf),
& iwdnpflw_for_slope(max_ncnls,max_n_weir_outf),
& isuppflw_for_slope(max_ncnls,max_nostr_cnl),
& isdnpflw_for_slope(max_ncnls,max_nostr_cnl),
& add_need(max_n_wcas,max_n_conv_cnl_wca),seepmon(max_n_lvsp_loc),
& total_outf_frm_cnl(max_n_wcas,max_n_conv_cnl_wca),
& reg_rel_frm_lok(max_n_eaa_condt),rindiv_canl_area(max_ncells,3),
& tneed_orig(max_n_wcas,max_n_conv_cnl_wca),
& avail_vol_pipel(max_n_eaa_condt), tneed_pipel(max_n_eaa_condt),
& avvolws(max_n_wcas,max_n_conv_cnl_wca),
& avvolrg(max_n_wcas,max_n_conv_cnl_wca),
& ibot_zone_regd(max_n_stor_w_oper_sch),daily_seep(max_n_lvsp_loc)

 DIMENSION  numd(12), rain(max_ncells), avol(max_ncnls),
& crmin0(max_ncnls,max_n_seas), volwtr(max_n_res),
& nbranch(max_ncnls), rstgpls(max_n_puls_rel),
& ws_bkp_cap(max_n_eaa_condt,2), crmin(max_ncnls,max_n_seas),
& wsuply(max_n_res), desgn_cap_canl_ws(max_n_eaa_condt),
& srsdmnd(366), stltrro(366), dmndro(8,2,2,366), prevqu(max_ncnls),
& s4runff(366), s65ernff(366), dmdsemind(366), uisrnff(366),
& runoff_south(max_n_eaa_condt), solmdph(max_ncells),
& runoff_avg(2), tinflow(max_n_wcas,max_n_conv_cnl_wca),
& ires_no(max_n_res),icanl_no_in_cell(max_ncells,3),
& res_outflow_to_basin(max_n_res,max_nostr_res),
& flow_from_asr_ws(max_n_res,max_nostr_res),
& flow_asr_inj(max_n_eaa_condt,max_n_reg_rel_basin),
& avg_sim_wca_stg_floor(max_n_wcas,max_n_minfllvl_loc),
& regcoffset(max_n_wcas), head_offset(max_ncnls),
& climate_index(12), tseep_divers(max_no_lvsp),
& tcnsrnff(366), tot_runoff_excess(max_n_eaa_condt), fplresws(366),
& outflow_from_3a(max_n_conv_cnl_wca), temp_volwtr(max_n_res),
& tempf_volwtr(max_n_res), rloc_seep_for_ws(max_n_serv_area),
& n_cells(max_n_gage_loc_outp), temp_weircff(max_n_res,
& max_nostr_res), tempf_weircff(max_n_res,max_nostr_res),
& rloc_excess_for_ws(max_n_serv_area), hdc0(max_ncnls,max_n_seas),
& rloc_excss_to_canl_next_step(max_ncnls), pfrunf(max_n_eaa_condt),
& total_reuse_ws(max_n_serv_area), icounty_loc_indx(max_ncnls),
& asr_rec_ws_cnty(max_n_serv_area), asr_rec_ws(max_n_serv_area),
& sumrain(max_ncells), gcell_mean_ponding_depth(max_ncells)
```

```
  DIMENSION pdsi(12), gcell_mean_ponding_depth_end(max_ncells),
& rchg_gw(max_ncells), qoutmx(max_n_res,max_nostr_res),
& pond_prev_gw(max_ncells), qoutmx_p(max_n_res,max_nostr_res),
& nodec(max_ncnlloc), depth_levee_spg_max_from_gw(max_ncells),
& sumrain_unadj(max_ncells), regstgwt(25), regstgwn(25),
& iregjulwt(25), iregjulwn(25), qhlws(max_n_eaa_condt),
& isuppflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
& rf_drawdwn_dph(max_ncnls,2,max_n_seas),
& isdnpflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
& ilvflw_for_slope(max_n_lvsp_loc,2),
& isuppflw_for_slope_kn(max_n_str_node_dep),
& isdnpflw_for_slope_kn(max_n_str_node_dep),
& hdrop_fact(max_n_cnls_w_var_slpe),
& hdrop_max(max_n_cnls_w_var_slpe),total_cell_stor(max_ncells),
& HWSEEP_F_L30(10),HDSEEP_F_L30(10),ilv_f_l30(10),
& icol_f_l30(10),irow_f_l30(10),ipos_lv_l30(10),
& sum_eaa_dmnd_ssm_nc(max_n_eaa_condt), rlosa_factor(8,2),
& diston_capac(2), diston_frac(2), diston_to_cal(366),
& diston_from_cal(366), diston_to_s4(366), diston_from_s4(366),
& caldmndro(2,366), s4dmndro(2,366), tcnsro(366),
& sem_cyp_tot_dmnd(366), sem_cyp_dmnd_local(366),
& hdrop_min(max_n_cnls_w_var_slpe),hist_lok_bndry_flw(11,366),
& rmin_ofml2(max_ncells),ofml2(max_ncells),
& reg_cap_fact(max_n_ostruc_wca,max_n_wcas,12),
& asr_t_cell_volm(max_ncells),pcbfrac(4),
& isupp_at_ups_end(max_ncnls,max_nostr_cnl),
& i_canal_rf_chg_ops(max_ncnls)

 CHARACTER*70 title_rain
 CHARACTER*200 infile, infil1
 CHARACTER bmp_opt*5, buffer*256, ccdmdro*6, cfcurfut*7,
& charid_asr(max_n_asr,max_src_dest_pts_for_any_asr)*6,
& convey_canal_names(max_n_tot_conv_cnl_wcas)*5, datetag*256,
& dmdronm*10, ds_convey_c_name(max_n_eaa_condt)*5,
& ds_int_canal_name_l8*5, eaa_conv_canal_names(max_n_eaa_condt)*5,
& env_ws_cutb_ssm_opt*3, idn_wca_name(max_n_eaa_condt)*5,
& idn_wca_name_env(max_n_eaa_condt)*5, idn_wca_name_l8*5,
& idowns_wca_name_lwdd*5, ieaa_res_asr_name(max_n_eaa_condt)*6,
& ieaa_res_asr_reg_name(max_n_eaa_condt,max_n_reg_rel_basin)*6,
& import_area_name(max_n_targ_area)*5, int_cnl_name_for_lwdd*5,
& name_res_for_inflow(max_n_div_to_ecp)*6, opt_bsn_prio_uncond*3,
& opt_prior_use_asr_flex_cal*3, res_file_name(max_n_sm_res)*80,
& rlist_lvname(max_no_lvsp)*5, rlv_seep_to_canl_c11b*5,
& runtitle*256, sdate*9, struc_name_meas(max_n_struc_meas)*6,
& use_ssm*3, weir_name(max_ncnls,max_n_weir_outf)*6, welldat_opt*5,
& wfld(max_n_wells)*9,fnm(max_n_str_node_dep)*6, bmp_ssm_opt*3,
& pcbruleopt*3, local_lwdd_struc_name*6

 INTEGER getfd, getfdd(150), ierr, ifltab1(1200), ifltabc(1200),
& ifltabr(1200), ifltabs(1200), ip(60),
& isum_nsm(max_n_rows_incl_lok), isum_w_lok(max_n_rows_incl_lok),
& maxx_nsm(max_n_rows_incl_lok), maxx_w_lok(max_n_rows_incl_lok),
& maxy_nsm, minx_nsm(max_n_rows_incl_lok),
& minx_w_lok(max_n_rows_incl_lok), rain_def

 REAL bflo_cap_c43est, bflo_cap_c44est, bflo_cap0_c43est(12),
& bflo_cap0_c44est(12), bflo_frac_c43est, bflo_frac_c44est,
```

```
     & bmp_makeup_water(366), bsn2est1(12,31), bsn2est2(12,31),
     & c43pet(366), c43rf(366), c44pet(366), c44rf(366),
     & cale_dmd_day(366), day_et_poten(max_ncells,31),
     & day_et_unsat(max_ncells,31), day_et_unsat_irr(max_ncells,31),
     & day_et_unsat_nonirr(max_ncells,31),
     & day_net_irr(max_ncells,max_n_type,31), day_pet0(max_ncells,31),
     & day_rain(max_ncells,31), dtime, etp(12), injection_max01,
     & injection_max02, iregjulwn, iregjulwt,
     & kflo0(max_n_struc_flw,366), kfloref(max_n_eaa_condt), lketvol,
     & lok_supp_c43est(12), lok_supp_c44est(12), lok_supp_day_c43est,
     & lok_supp_day_c44est, lok_targ_level, istokabdmdnotmet,
     & lok_targ_level_wet1(max_n_goal_ssm),
     & lok_targ_level_wet2(max_n_goal_ssm), lok2caloos_est,
     & lok2caloos_irrig, lok2stluc_est, lok2stluc_irrig, loketvol,
     & lvseep_to_c6, lvseep_to_l33, mod_delta_stor(366),
     & mon_pet0(max_ncells), norstorpet(366), norstorrf(366), qin_max01,
     & qin_max02, qout_max01, qout_max02, rfp(12),nlsdmndro(2,366),
     & rinit_wca_stage(max_n_wcas), sle_dmd_day(366), timearray(2),
     & timediff, withdrawal_max01, withdrawal_max02, wup(12), wup0(12)

      DOUBLE PRECISION CRMIN,CRMIN0,crmin_bel_flr,hdc0


      DATA b /'
     &
     &
     &                                                          '/
      DATA numd /31,28,31,30,31,30,31,31,30,31,30,31/

      EXTERNAL getfd
C
C
C
      timediff = dtime(timearray)
C
C   Check for existence of environment variable that
C   sets the data directory path for output files to
C   be stored in.  "SFWMMDAT" = directory name for output files.
C   If environment variable not set prior to execution of the model,
C   the current directory is assumed to be the path for output.

      CALL GETENV("SFWMMDAT",DATADIR)

      IF (DATADIR(1:5) .EQ. '     ') THEN
         DATADIR = "."
      ENDIF

C   OPEN_INPUT_DATA_FILES
C   Open input data files required by the model (names of files
C   input in ALTWMM file). Defines DMNDRONM; CCDMDRO; and
C   CFCURFUT variables.

      CALL OPEN_INPUT_DATA_FILES(RUNTITLE,INFILE,INFIL1 ,dmdronm,
     $ccdmdro,cfcurfut,ifltab1,ifltabr)

C   GEN_MODEL_RUN_DEF_PARAM
C   Input run definition parameters. TYPERUN; RUNMODE, USE_LECT_ET,
```

```
C   USE_TRIGGER,etc. Then input min & max column for each row of grid
C   system. Determine isum array for NSM grid rainfall for mapping NSM
C   rain into SFWMM grid network.

      CALL GEN_MODEL_RUN_DEF_PARAM(xspc,yspc,ibm_initc,npopts,
     $maxy_nsm,minx_nsm,maxx_nsm,isum_nsm,isyr,ip,isyretbin,
     $iseas0)

C   GEN_MODEL_DEF_PARAM_DATA
C   Input general definition data for areas (EAA, WCAs, etc.) in model
C   domain. Input # of canals simulated, canal locations, etc.  Store
C   all canal nodes in NODECN array and NPTS keeps track of where they
C   are.  Also input data for canals whose head drop will vary daily.
C   Read in runoff and rainfall nodal run definition parameters.  Read
C   in names of structures with measured flows [STRUC_NAME_MEAS] and
C   names of canals [CNLNAME] desired to be output. Read in master list
C   of names of structures based on indexing [FLNM]. Read in landuse
C   information.  Read in definition parameters for LOK and operational
C   parameters for EAA.  Read in RES/ASR parameters for EAA basins.Then
C   read canal related information.  Read in structure capacities.

      CALL GEN_MODEL_DEF_PARAM_DATA(ifacteaa,isem_flg ,
     $max_days_mean_loktasr,max_days_mean_bkpump,rf_factor ,ro_factor,
     $capac_sugh,lok_targ_level,use_ssm ,import_area_name,IP,npopts,
     $nsvarea,nlu,capacc10a_bak ,desgn_cap_canl_ws,welldat_opt,NTSTRCTR,
     $env_ws_cutb_ssm_opt ,bmp_opt,opt_bsn_prio_uncond,
     $opt_prior_use_asr_flex_cal ,frac_thru_s8_sta34,n_cells,
     $res_seep_factor,ds_convey_c_name ,ieaa_res_asr_name,
     $ieaa_res_asr_reg_name,name_res_for_inflow ,eaa_conv_canal_names,
     $base_area_eaa_basin,idn_wca_name ,idn_wca_name_env,depth_soil_eaa,
     $solinit,icol_stg,irow_stg ,int_cnl_name_for_lwdd,
     $ds_int_canal_name_l8,idn_wca_name_l8 ,idowns_wca_name_lwdd,
     $fracdph_min,fracdph_max, ssmminfrac,lok_targ_level_wet1,
     $lok_targ_level_wet2,issm_cutb_opt_bcyp,
     $issm_cutb_opt_brghton,rlok_stg_beg_forpmp,rlok_stg_end_forpmp,
     $iregcals79opt,ws_bkp_cap,MAXBPCNTR,rf_factor_lok,ro_factor_lok,
     $nwgoalsto,mon_targ_wet,iday_targ_wet,struc_name_meas,
     $iwsbkpwsta,bflo_frac_c43est,bflo_cap_l8,frac_dmnd_met_wt,
     $frac_dmnd_met_wn,rmean_cnl_bot_elev,flow_resistvty_coef,
     $rf_drawdwn_dph,hdrop_fact,hdrop_max, no_canl_in_cell,
     $icanl_no_in_cell,ipos_cell_in_canl, rlosa_factor, s235_capac,
     $s235_frac, diston_capac, diston_frac, semirr_dmnd_factor,
     $fpl_dmnd_factor,hdrop_min,bmp_ssm_opt,pcbruleopt,pcbfrac,
     $frac_cal_dmnd_met_by_del,allig_alley_res_fact,
     $iroute_l8rnf_to_sta_opt)
c
c   LEC_ET_INITIAL_READ
C   Open files for LEC_ET module. Opens files used for the LEC-ET
C   calculation and reads appropriate data.  Open and initialize the
C   grid output files for unsaturated, UNSAT_IRR, UNSAT_NONIRR, ponding
C   and saturated ET. If trigger module is active, open and initialize
C   output file forshortages. [Then for estimated ET - used for LECSA
C   processing].
c
      IF(use_lec_et .ge. 1) THEN
         call lec_et_initial_read(ism,isyretbin,ibm,getfdd,datadir)
      ENDIF
```

```
c
c   OPER_SCHED_DATA
C   Input data for operational schedules for LOK and WCAs, Holeyland,
C   and Rotenberger Tract.  Any proposed calendar based operational
C   schedule for additional reservoirs may be input.  Includes global
C   climate indicators in operation options and flood control & pulse
C   zones. Read in estimatedwater use for LOSA for determination of
C   supply side management cutbacks.Read in projected ET for LOK for
C   1, 3, 6 months determination.
c
      CALL OPER_SCHED_DATA(clim_threshold_est,offset_to_sched
     $, wup0,wup,numd,ssm_vol_cutback_thres,month_end_credit
     $,nbrkpt_ssm1,iopt_for_semcyp_prior_roten,ibeg_mth_lokdev
     $,iend_mth_lokdev,ibeg_day_lokdev,iend_day_lokdev,etp,rfp
     $,iopt_drawdown,regstgwt,regstgwn,nbrkpt_ssmwt,nbrkpt_ssmwn
     $,iregjulwt,iregjulwn,ibot_zone_regd)
c
C   RESERV_INPUT_DATA
C   Input physical and operational data for reservoirs/STAs simulated
C   including LOK outlet structures; input file = reservoir_input.dat.
C   Defines reservoirs to be maintained at desired minimum levels during
C   dry period.  Includes data for inflow and outflow structures.  Also
C   reservoir inflow and outflow structure names.
c
      CALL RESERV_INPUT_DATA(ires_no,ieaa_res_asr_name ,
     $ieaa_res_asr_reg_name,name_res_for_inflow ,no_of_res_reg_frm_lok,
     $import_area_name ,ntotal_no_import_areas,n_add_lev_sys,
     $no_of_small_res_budg_files,iunit_no_res_file,
     $res_file_name,nrgtrg,noder_trg,stgr_trg,aqperm_in_res,
     $AQDEP_in_res,RCPFACTEAA,iopt_env_eaar_thru_g372)

C
C   LOK_O_WCA_IN_STRUC_DTA
C   Input physical and operational data for LOK outlet and WCAs inlet
C   structures. Computes discharges for structures that are dependent on
C   stages at one or group of grid cells. Only structures discharging
C   from areas outside the WCAs (S-8, S-10A, S-3) are included.
C   NOTE: Input structure names simulated in STRUCTURE_LOK_IN_WCA_FLW.F.
c
      CALL LOK_O_WCA_IN_STRUC_DTA(ires_no,idshg_opt,noutlets ,
     $istrctr_ws,nres_ws,istr_count,struc_name_meas)
c
C   WCAS_INPUT_DATA
C   Physical and operational data for WCAs and WCA outlet structures.
C   Open I/O files and retrieve physical data required by the model.
C   Input WCA names, initial stages, basin #, option for rain driven
C   operations for each WCA. Conveyance canals used as triggers for
c   outflow for each WCA.
c
      CALL WCAS_INPUT_DATA (ds_convey_c_name,int_cnl_name_for_lwdd ,
     $ds_int_canal_name_l8,idn_wca_name,idn_wca_name_env ,
     $idowns_wca_name_lwdd,idn_wca_name_l8,nconvey_total ,
     $convey_canal_names,rinit_wca_stage,n_conv_tot,import_area_name ,
     $eaa_conv_canal_names,ntotal_no_import_areas,istr_count,
     $g94abcapac,regcoffset,iopt_lim_des_cap_wca,isdnpflw_for_slope_wca,
     $isuppflw_for_slope_wca,reg_cap_fact)
C
```

```
C   ASRINPUT
C   Input data and set parameters for ASR wells. Read in parameters
C   for each bubble. Each ASR has an efficiency factor defined as
C   Eta(eff. factor)=volume recovered/volume injected.  ASRs have
C   multiple sources/destinations (canals, reservoirs, EAA, RO, etc).
C   For each ASR, the following has to be defined - for injection and
C   recovery.  Input ASR injection and recovery capacity. Also 3 of
C   canal sources, canal name, trigger stage in canal for injection into
C   ASR for each canal source. Number of ASR systems in model, flag for
C   Caloosahatchee & St Lucie basin/reservoir, option to use ASR to meet
C   Caloosahatchee & St Lucie estuarine requirements.
c
      IF (runmode.eq.'SIMUL') call asrinput(iuse_asr_env1,iuse_asr_env2,
     $charid_asr)
c
C   OPEN_ASCII_OUTPUT_FILES
C   Open ASCII files for output purposes.  Basin budget components,
C   reservoir budget components, ASR budget components, summary of
C   inflows to STAs, canal budget summary, Caloosahatchee and St Lucie
C   basin/ reservoir components. Set up minimum flows and levels file,
C   daily import stage for Everglades.  Reads Palmer Drought Severity
C   Indices, summary of simulated heads for variable slope canals,
C   structure inflows intoreservoirs recharging groundwater, WSE LOK
C   operations, etc.
c
      CALL OPEN_ASCII_OUTPUT_FILES(NUMD,ISYR
     $, IP,XSPC,YSPC,nlu,welldat_opt,ntotres,icol_stg,irow_stg
     $, no_pred_storms_thres_wca,itrib_hydro_cond_opt,trib_rf_et
     $, s65e_runff_wkly,no_of_small_res_budg_files
     $, iunit_no_res_file,res_file_name,ncnl_dly_slope
     $, canal_name_dvslope,ntotasr,cal_asr_flag,stl_asr_flag
     $, charid_asr)
c
C   GRID_CELL_BASED_DATA
C   Input grid cell based data from STATDTA file. Input land surface
C   elev(ells), storage coefficients, land use nodes, basin identifiers
C   and nodes, initial gw elevation.  Set EAA operational parameters and
C   initial conservation area stages.  Read aquifer depth levels and
C   permeability data, transmissivity, ET basin identifiers (IETZON),
C   infiltration rate, irrigated areas, fraction of irrigated area in
C   each cell, effective root zone depth, max soil moisture holding
C   capacity in unsaturated zone and fraction of landscape receiving
C   water from PWS wells.
c
      CALL GRID_CELL_BASED_DATA(sumrain,TAREAEAA,SOLMDPH,pond0 ,solmx0,
     $hinit,rinit_wca_stage,base_area_eaa_basin ,ibasin_no_wca,
     $grid_area_res,depth_soil_eaa ,initialize_annually_opt,
     $solinit ,fracdph_min,fracdph_max,n_add_lev_sys,total_afact,
     $no_nonscane_irr_cells_l8,ofml2,rmin_ofml2,total_area_298d_grid)

c
C   OPEN_BINARY_OUTPUT_FILES
C   Open binary files (including DSS) for output purposes.
C   Sets GETFDD array - usually based on IP flags, also calls GRIDWHD
C   basedon IP flags. GETFDD sent to PRINTLP but does appear to be used.
c
      CALL OPEN_BINARY_OUTPUT_FILES(RUNTITLE,getfdd,ip,isum_nsm
```

```
     $,minx_nsm,maxx_nsm,maxy_nsm,isum_w_lok,minx_w_lok
     $,maxx_w_lok,maxylok,numd,use_lec_et,ibm,ism,isyr
     $,ifltabs,ifltabc,rain_def,title_rain)
      prev_avg_lokstg = stagelo
C
C   CNLDATA
C   Input data for the canals and corresponding outlet structures.
C   Open and read canal data files.  Input canal and structure
C   definition data to be used in CHNLF.F.  Input data for outlet
C   structures simulatedas fixed crest weirs.  Then names of structures
C   with downstream nodalstage targets as limit to flow.  Then data
C   required to simulate discharges through additional outlet
C   structures.  Other files include known flow specification data,
C   restart data, levee specifications for calculation of levee seepage,
C   storm data, drawdown for reservoirs/outlet structures,data for
C   canals with dual operations as a function of discharges elsewhere.
C   Output with levee seepage, daily canal stage data, etc.
c
      CALL CNLDATA(CRMIN,NBRANCH,istr_count ,
     $ires_no,rlist_lvname ,import_area_name,rlv_seep_to_canl_c11b ,
     $icounty_loc_indx,n_cells,CRMIN_bel_flr,crmin0,ndsglim ,
     $nodedlim,fl_trig_constrt,month_beg_min_mod_flr,
     $month_end_min_mod_flr,iday_beg_min_mod_flr,iday_end_min_mod_flr,
     $node_wd,hdc0,icnl_no_reg_seas,iweir_no_reg_seas,nreg_seas,
     $weir_reg_seas,ivary_slope_daily_opt,mid_point_canl,
     $iwuppflw_for_slope,iwdnpflw_for_slope,isuppflw_for_slope,
     $isdnpflw_for_slope,ilvflw_for_slope,isuppflw_for_slope_kn,
     $isdnpflw_for_slope_kn,is334_open_flg,frac_avail_cap_s333,
     $frac_avail_cap_s334,weir_name,s333s334rg_offset,
     $rindiv_canl_area,no_canl_in_cell,fnm,
     $n_gated_grav_outlets_fc,n_pump_outlets_fc,rmax_dschg_coef,
     $isupp_at_ups_end,iroute_acme_fc_to_sta1e_opt)

c
C   TRIGGER_INPUT
C   Input data for trigger module - read cutbacks for PWS and industrial
C   as a fraction of total required. Cutback for other suppliers is in
C   max inches per month.  Also min days LOK in SSM.  Then location,
C   trigger & heads of four phases, length of previous month cutback for
C   each zone in each service area section.
c
      IF(use_trigger .ge. 1) THEN
         lec_restrictions = .FALSE.
         lec_restrictions1 = .FALSE.
         call trigger_input(use_trigger, nch, min_lok_ssm_cnt)
      ENDIF
c
C   INIT
C   Initialize variables: structure indices, basin id #, service area
C   ID #, reference lake stage array for pulse releases from LOK to
C   St Lucie & Caloosahatchee.  Initialize volume of water STAs can
C   receive; 0 if they do not exist.
c
      CALL INIT(VOLWTR,RSTGPLS,rlist_lvname ,import_area_name,NTSTRCTR,
     $numd,nyday ,runtitle,wsuply,tot_cnl_dmd_asr ,
     $outflow_to_canal_roten,outflow_to_marsh ,flow_to_sta3_fr_298_nnrc,
     $flow_to_sta2_fr_298d ,stagelo_res,lok_split_option,nmdays_lok ,
```

```
     $climate_index,no_of_hurr_pred,tot_runoff_excess ,hres01,asr_vol01,
     $hres02,asr_vol02,itypern,IWSBPCNTR,NCA_R,s356_to_cell,
     $s356_to_l29,s333_reg_lec,s337fc,s334_fc,iroute_to_wca3a_frm_holy,
     $total_lec_dmnd_lok,tot_avl_l8_bpit_res_m1q,sum_eaa_dmnd_ssm_nc,
     $prevqu,c304_out_to_marsh,sum_eaa_dmnd_ssm,sum_alocnnr,
     $ws_from_lo_s6,rnlkblt_to_s29,dellec,sum_alloc_ssm,sum_losa_dmnd,
     $igravity_s8_nosta,igravity_s7_nosta,tempf_weircff,temp_weircff,
     $flow_to_c9r_frm_c11,ws_frm_lok_to_semcyp,
     $dischg_to_bay_s29_frm_c11r,rloc_excss_to_canl_next_step,
     $istokabdmdnotmet,pctmet,flow_to_sta1w_fr_298d,
     $out_flow_east,out_flow_east0)


c
C--------------------------------
C     TIME ITERATION STARTS HERE
C--------------------------------
      DO 60 IMTH = IBM,IEM
         IYEAR = (IMTH - 1)/ 12+ 1
         IYR = IYEAR + IFYR - 1
         MONTH = IMTH - (IYEAR - 1) * 12
c
         IF (month .eq. ibm_initc .and. mod(iyear-1,ifreq_init) .eq. 0)
     $   THEN
C   Option to reinitialize LOK stages, and water levels during
C   simulation
            IF (initialize_annually_opt .eq. 1) THEN
               DO I = MAXY,1,-1
                  DO NODE = ISUM(I)+1,ISUM(I+1)
C   For EAA basins
                     IF ((runmode .eq. 'SIMUL' .and. IBSN(NODE) .GE.  7.
     $               and. IBSN(NODE).LE.  10.AND. land_use_type(LUTYP
     $               (NODE)) .NE. 'WETLAND') .or.IBSN(NODE).EQ. 40.OR.
     $               IBSN(NODE).EQ.10.OR.IBSN(NODE).EQ.42) THEN
                        SOLMX(NODE) = SOLMX0(NODE)
                     ENDIF
                     total_lec_dmnd_lok = 0.0
                     do ijk=1,max_n_eaa_condt
                        sum_eaa_dmnd_ssm_nc(ijk) = 0.0
                        sum_eaa_dmnd_ssm(ijk) = 0.0
                     enddo
                     tot_avl_l8_bpit_res_m1q = 0.0
                     H(NODE) = HINIT(NODE)
                     POND(NODE) = POND0(NODE)
                     SUMRAIN(NODE) = 0.0
                     SUMET (NODE) = 0.0
                     SUMRAIN_unadj(NODE) = 0.0
                     SUMET_outside_res(NODE) = 0.0
                     adjust_dur_excess(node) = 0.0
                     adjust_dur_deficit(node) = 0.0
                     adjust_dur_deficit0(node) = 0.0
                     prev_adjust_for_area(node) = 0.0
                     chg_depth_levee_spg(node) = 0.0
                     daily_excess_adjust(node) = 0.0
                     adjust_for_area(NODE) = 0.0
                     adjust_for_area_n(NODE) = 0.0
                     adjust_for_area_m(NODE) = 0.0
```

```
                adjust_for_area_m1(NODE) = 0.0
                adjust_for_area_m2(NODE) = 0.0
                adjust_for_area_m3(NODE) = 0.0
                adjust_for_area_m4(NODE) = 0.0
                sum_chg_depth_levee_spg(NODE) = 0.0
                sum_chg_depth_levee_spg_unadj(NODE) = 0.0
                sum_chg_dpth_ovl_flw(NODE) = 0.0
                outf_fc_frm_res_to_cell(NODE) = 0.0
                outf_seep_loc_to_res_cell(NODE) = 0.0
                seep_loss_res(node) = 0.0
                seep_loss_res_cell(node) = 0.0
                daily_outf_from_res_to_cell_max(node) = 0.0
                daily_outf_seep_loc_to_res_cell(node) = 0.0
                daily_outf_fc_frm_res_to_cell(node) = 0.0
                pond_adj_due_to_res(node) = 0.0
                istokabdmdnotmet = 0.0
                DO ir = 1,noresincell(NODE)
                   iresnum = ires_index_in_cell(node,ir)
                   IF(rinit_res_stg(iresnum) .lt. -900.) THEN
                      stage_res(node,ir)=H(NODE)+POND(NODE)
                      pond_in_res(node,ir)=amax1(stage_res(node,ir)
$                     - ells_in_res(node,ir),0.0)
                   ELSE
                      stage_res(node,ir)=rinit_res_stg(iresnum)
                      pond_in_res(node,ir)=amax1(stage_res(node,ir)
$                     - ells_in_res(node,ir),0.0)
                   ENDIF
                   prev_stage_res(node,ir) = stage_res(node,ir)
                ENDDO
             END DO
          END DO
          tot_avl_l8_bpit_res = 0.0
          tot_cnl_dmd_asr = 0.0
          flow_to_sta3_fr_298_nnrc = 0.0
          flow_to_sta2_fr_298d = 0.0
          total_outflow_from_roten = 0.0
          total_outf_from_holy_g200a = 0.0
          outflow_from_roten_to_west = 0.0
          outf_to_c9res_via_us27s = 0.0
          outflow_from_roten_to_wca = 0.0
          outflow_from_roten_to_holy = 0.0
          sta34_out_to_s140 = 0.0
          sta34_out_to_wca3ane = 0.0
          outflow_from_sta6_to_s140 = 0.0
          rem_inflow_to_meet_holy_sched = 0.0
          outflow_from_sta6 = 0.0
          outflow_to_canal_roten = 0.0
          outflow_to_marsh = 0.0
          outf_from_c11res = 0.0
          flow_to_c9r_frm_c11 = 0.0
          ws_frm_lok_to_semcyp = 0.0
          dischg_to_bay_s29_frm_c11r = 0.0
          do i = 1,max_ncnls
             rloc_excss_to_canl_next_step(i) = 0.0
          enddo
          c304_out_to_marsh = 0.0
          sum_alocnnr = 0.0
```

```
                ws_from_lo_s6 = 0.0
                rnlkblt_to_s29 = 0.0
                dellec = 0.0
                sum_alloc_ssm = 0.0
                sum_losa_dmnd = 0.0
                igravity_s8_nosta = 1
                igravity_s7_nosta = 1
                DAY1 = .TRUE.
                DO i = 1,ntotres
                   wsuply(i) = 0.0
                   rinflw_to_res_frm_cnl(i) = 0.0
                   res_struc_inflw(i) = 0.0
                   DO kk = 1,max_nostr_res
                      tempf_weircff(i,kk) = 99999.
                      temp_weircff(i,kk) = 99999.
                   END DO
                END DO
                DO n = 1,nch
                   chdep(n) = chdep0(n)
                   prevqu(n) = 0.0
                   DO kk = 1,2
                     hdc(n,kk) = hdc0(n,kk)
                   ENDDO
                   DO kk = 1,nstrctr(n)
                      DO ns = 1,nsubstr(n,kk)
                         lpump(n,kk,ns)=.false.
                         lgrav(n,kk,ns)=.false.
                         LPUMPF(n,kk,ns)=.false.
                      ENDDO
                   ENDDO
                END DO
                if (ntotres .gt. 0) then
                   DO ia = 1,ntotres
                     IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &               (ia) .eq. 'NO ') THEN
C
C   LARGER_RESERV_STOR
C   step Calculate available storage for flow into reservoirs for next
C   time and determine the maximum available storage for each reservoir.
C   It uses stage & ells for each model node not just for the reservoir
C   to initially  calc VOLWTR.
C

                       CALL larger_reserv_stor(IA,VOLWTR,SUMRAIN,QOUTMX
     &                 ,total_afact,deltastor,prev_stage_res,tvolwtrl
     &                 ,iseas0)
                     ELSE
C
C   SMALL_RES_STOR
C   Determine maximum available storage for each small reservoir treated
C   as a separate entity in the model grid domain.  This routine does
C   not adjust for overland flow but it does in the outflow structures.
C   This is based onres_out_type which is for each reservoir & outflow
C   structure, the purpose of outflow through each structure.  If true
C   then QOUTMX is calculated  andadded to volwtr ELSE QOUTMX = 0.0.
C   Processed at the beginning of each time step and before
C   SPECSETUPFDSS.
```

```
C
                      CALL small_res_stor(IA,VOLWTR,QOUTMX,sumrain,
     &                total_afact,prev_stage_res,deltastor,tvolwtrl,
     &                avvolgr_mx,iseas0)
                    ENDIF
                 ENDDO
               endif
               stagelo = stagelo0
               ipulse = 0
               IWSBPCNTR = 0
               nmdays = 0
               ndays_sim = 0
               nmdays_lok = 0
               iroute_to_wca3a_frm_holy = 0
               NCA = 0
               NCA_OV = 0
               NCA_ALT = 0
               NCA_R = 0
               nmdays_rainavg = 0
               nmdays_rainavg0 = 0
               s356_to_l29 = 0.0
               s356_to_cell = 0.0
               s333_reg_lec = 0.0
               s337fc = 0.0
               s334_fc = 0.0
               hres01 = 0.0
               asr_vol01 = 0.0
               hres02 = 0.0
               asr_vol02 = 0.0
               HOLPUMP = .FALSE.
               LS151 = .TRUE.
               DO i = 1,10
                  rstgpls(I) = 0.0
               ENDDO
               DO i = 1,neaabsn
                 tot_runoff_excess(i) = 0.0
               END DO
               do i = 1,5
                 LFORPUMP(i) = .FALSE.
               enddo
               do i = 1,nwcas
                 LFRACBFLR(i) = .FALSE.
               enddo
               do i = 1,2
                 runoff_avg(i) = 0.0
               enddo
               DO i = 1,40
                 avg_sim_wca_stg_out(i) = 0.0
               ENDDO
            ENDIF
         ENDIF
C   Option to reinitialize LOK stages, and water levels during
C   simulation is turned off.
         IF (DAY1.and.initialize_annually_opt .eq.0) THEN
           if (ntotres .gt. 0) then
              DO ia = 1,ntotres
                 IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
```

```
     &           (ia) .eq. 'NO ') THEN
C
C   LARGER_RESERV_STOR
C   Calculate available storage for flow into reservoirs for next time
C   step and determine the maximum available storage for each reservoir.
C   It uses stage & ells for each model node not just for the reservoir
C   to initially calc VOLWTR.
C

                   CALL larger_reserv_stor(IA,VOLWTR,SUMRAIN,QOUTMX
     &             ,total_afact,deltastor,prev_stage_res,tvolwtrl
     &             ,iseas0)
                 ELSE
C
C   SMALL_RES_STOR
C   Determine maximum available storage for each small reservoir treated
C   as aseparate entity in the model grid domain.  This routine does not
C   adjustfor overland flow but it does in the outflow structures. This
C   is based on res_out_type which is for each reservoir & outflow
C   structure, the purpose of outflow through each structure.  If true
C   then QOUTMX is calculated and added to volwtr ELSE QOUTMX = 0.0.
C   Processed at the beginning of each time step and before
C   SPECSETUPFDSS.
C
                   CALL small_res_stor(IA,VOLWTR,QOUTMX,sumrain,
     &             total_afact,prev_stage_res,deltastor,tvolwtrl,
     &             avvolgr_mx,iseas0)
                 ENDIF
               ENDDO
           endif
         ENDIF
c
         IF ((IMTH.EQ.IBM).OR.(MONTH.EQ.1)) THEN
c
c   Input data for LOK inflows and other boundary inflows.
c   climate variables for LOK operations input also.
c

          IF (runmode .eq. 'CALIB') THEN
C   HIST_LOK_BNDRY_INPUT_DATA
C   Read in DSS file only if RUNMODE is calibration.  Extract history
C   boundary condition and modified delta storage & S-65E, Taylor
C   Creek/Nubbin Slough, FLP reservoir flows.  Demands of Seminole
C   Indians on north side of LOK. Also one year of monthly ET and
C   rainfall data for LOK via FC=3.
C
            CALL hist_lok_bndry_input_data(imth, ibm, ifyr, iyear,
     $      istr_index, kflo0, mod_delta_stor, s65ernff, tcnsrnff,
     $      fplresws, hist_lok_bndry_flw, dmdsemind, lp, nday_yr,
     $      numd, nflpts, ifltabr, uisrnff)
          ELSE
C
C   LOK_BNDRY_INPUT_DATA
C   Input data for LOK inflows and other boundary inflows/climate
C   variables for LOK operations.  Also input climate variables for
C   WSE schedule for LOK and  WCA operations.  Read modified delta
C   storage for LOK.  Read flows through S-65E into LOK, flows to FPL
```

```
C   reservoir.  Makeup water based on BMP includescontrolling RO to
C   reduce pollutants in the EAA by retaining water in drainage ditches,
C   water levels, etc.  Read in BMP makeup water according to BMP makeup
C   rule for LOK.  Demands of Seminoles on north side of LOK. Read in
C   estuary demands.  This year's daily C43/C44 basin rainfall * daily
C   north storage res rainfall.  Major EAA basin.  Read 1 year of
C   monthly ET and rainfall data for LOK.
C
            CALL LOK_BNDRY_INPUT_DATA(imth,ibm,ifyr,iyear,
     $      climate_index ,no_of_hurr_pred,rmulti_seas_pred,
     $      iatl_convyr_index,istr_index ,kflo0,mod_delta_stor,s65ernff,
     $      tcnsrnff,dmndro,stltrro ,s4runff,bmp_makeup_water,fplresws,
     $      dmdsemind,srsdmnd ,srsdmnd_w,srsdmnd_e,sle_dmd_day,
     $      cale_dmd_day,c43rf,c44rf ,norstorrf,sem_cyp_dmnd,
     $      diff_runoff_bmp,frac_area_eaa_basin ,isem_flg,lp,nday_yr,
     $      ro_factor,dmdronm,numd,morefet,lokmorn ,ccdmdro,cfcurfut,
     $      rf_factor,rlosa_factor,itribres2,idbsnopt1 ,idbsnopt2,
     $      sle_dmnd_factor,cale_dmnd_factor,inorth_stor_opt ,
     $      frac_sem_cyp,bmp_opt,nflpts,ifltab1,ifltabr,pdsi ,
     $      diston_capac, diston_frac, diston_to_cal, diston_from_cal,
     &      diston_to_s4, diston_from_s4, caldmndro, s4dmndro,
     &      nlsdmndro, tcnsro, semirr_dmnd_factor, fpl_dmnd_factor,
     &      sem_cyp_tot_dmnd, sem_cyp_dmnd_local, uisrnff)
          ENDIF
C
C   ACCUM_ESTUAR_DMNDS -
C   Accumulate estuarine demands for all months of current year.
C   Only used if estuarine accounting is set to daily frequency.
C   Calculates LOK supplemental release and max backflow capacity
C   for all months of current year. Set up daily & monthly estuarine
C   demands based on simsles & simcaes options.
C
            IF (es_dmnd_acc_freq .ne. 'DLY') CALL ACCUM_ESTUAR_DMNDS
     $      (numd,bflo_cap0_c43est ,lok_supp_c43est,lok_supp_day_c43est,
     $      lok_supp_c44est ,bflo_cap0_c44est,bsn2est1,bsn2est2,
     $      cale_dmd_day ,sle_dmd_day,stltrro,dmndro,ifyr,iyear)

c
C   MUNIC_WELL_PUMP_SETUP
C   Input municipal and industrial - residential self supplied
C   withdrawalsand set up necessary arrays. Note if multiplier of
C   municipal wellfieldpumps not supplied then warning is printed and
C   multiplier is set to 1.0.The format for reading FC=18
C   [WELL_PUMPAGE_061303_ASR.DAT] is based on WELLDAT.OPT.  Then FC=37
C   [pumpage for industrial and residential, self-supplied pumpage for
C   each month] is read.
C
            CALL MUNIC_WELL_PUMP_SETUP(welldat_opt,WFLD)
c
C   First call to SIMQ2DSS is just to initialize arrays.
C
C   Initialize arrays - writes daily simulated structure discharges into
C   output DSS file 'STR2x2.DSS' -- Overwrites.
C   NOTE: STRFLW{structure flow points} is initialized for each day and
C   for each structure.  Also processed at end of monthly time step
C   after sum_trig_heads.F.
C
```

```
c   SIMQ2DSS -Initializes arrays here - does not write here.
c
            IF(IMTH.EQ.IBM) CALL SIMQ2DSS(IFLTABS,SDATE,IMTH)
C
C   This subroutine dumps daily flows from simulated structure into
C   the DSS file 'STRFLOW.DSS'; if pathnames already exist, it writes
C   over them; if not, it appends these simulated flows at the end of
C   the DSS file.
C
C   SIMCSTGDSS - Initializes arrays - does not write here.
C
            IF(IMTH.EQ.IBM) CALL SIMCSTG2DSS(IFLTABC,SDATE,IMTH)
            CALL YMDDAT(IYR,MONTH,1,104,SDATE,NDATE,IERR)
            IDSS=0
c
c   ANNUAL_INIT
c   Initialize variables that are accumulated annually.
c
            CALL ANNUAL_INIT(lhls,iyear,runmode,ifyr)
C
         ENDIF
         IF (MOD(IMTH,12) .EQ. 1) THEN
            NYDAY = 0
         ENDIF
c
c   MTHLY_VARIABLES_INIT
c   Initialize variables that are accumulated monthly.
c
         CALL MTHLY_VARIABLES_INIT(rmonthly_loss_bmp,RMETVOL,
     $   rmetvol_res,lok_ssm_cnt,crmin,mon_pet0,
     $   rmthly_rf_lok,crmin0,CRMIN_bel_flr,innrc,
     $   ofml2,rmin_ofml2,owpond2,owpond,asr_t_cell_volm)
c
C
C   RAIN_PET_SETUP
C   Input data plus set up rainfall [DAY-RAIN is mapped to
C   DAY_RAIN_NSM].  PET arrays - map rainfall array for NSM model into
C   SFWMM grid network.  Monthly rainfall is used later for ET
C   calculations. Read one month of daily PET (referenced to grass) for
C   each 18 new ET basins into PETSTN array. Then calculate reference
C   ET for each node (potet array) based on weights.  POTET data summed
C   monthly to be used later for ET reductions.
c
         CALL RAIN_PET_SETUP(day_rain,maxy_nsm,minx_nsm,maxx_nsm,isum ,
     $   dx,dy,itrib_hydro_cond_opt,numd ,mon_pet0,day_pet0,
     $   potet,c43pet,c44pet,norstorpet ,day_pet0_lok,
     $   pet_weight,month,maxy,netzon,nday_yr ,rain_def,title_rain,
     $   isum_nsm,day_rain_nsm)
c
C
C   MTHLY_UNSAT_ET_IRR_COMP
C   Compute monthly ET and irrigation in LEC developed areas using
C   output from AFSIRS for the current month.  Reads one month of daily
C   ETU components and net irrigation demands using LEC_ET_TIME_SERIES_
C   READ routine. Compute monthly sums for only the net irrigation
C   demands and for the unsaturated zone ET from irrigation areas.  The
C   monthly sumsfor the actual ET and supply components, etc will be
```

```
C   computed and written to binary input via LEC_ET_SUM_MONTHLIES.  The
C   monthly net irrigation demands and ET_UNSAT_IRR are needed for
C   computing reductions in ET via the EST_ETIU function. Routine reads
C   3 ET components (pet,etu, etiu) via LEC_ET_TIME_SERIES_READ.F, then
C   sets daily et components; and reads 6 net irrigation components
C   (landscape, nursery, golf,ag_lowvol, ag_overhd, ag_other) via
C   LEC_ET_TIME_SERIES_READ.F.
c
         IF(use_lec_et.ne.0) THEN
            call mthly_unsat_et_irr_comp (numd,day_net_irr,
     $        day_et_unsat_irr,day_et_poten,day_et_unsat_nonirr,
     $        day_et_unsat,month,wfld,pump_def_semhlywd,
     $        pump_semhlywd)
C
C   LEC_PWS_IRRIG
C   Compute monthly ET and irrigation supplies & shortages in LEC
C   developed areas using output from AFSIRS for current month.  Call
C   REDUCE_WELLQ to reduce water supply pumpage if trigger module is to
C   be used and if LECwater restrictions are in effect for current
C   month [use TRIGGER=1 & LEC_RESTRICTIONS='T'].  Compute monthly
C   ETIU_UNRESTR & ETIU_RESTR and public water supply and shortage as
C   well as update well pumpage array in  LEC developed areas. Compute
C   public water supplies and shortages.  Compute irrigation supplies
C   and shortages.  If there is a shortage for the current irrigation
C   use type, then compute the estimated reduction in unsaturated zone
C   ET using regression equations developed to mimic AFSIRS.
C
            call lec_pws_irrig (mon_pet0,month,iyr,wfld,
     $        pump_def_semhlywd, pump_semhlywd)
         ENDIF
c
C----------------
C   BEGIN DAILY TIME STEP ITERATION
C-----------------
         IDYMAX = NUMD(MONTH)
         DO 50 IDAY = 1,NUMD(MONTH)
c
C   DAILY_VARIABLES_INIT
c   Initialize daily variables.
C   Init available capacity for injection into and recovery from ASR
C   wells; modified delta storage; makeup water; runoff reduction in EAA
C   basin due to BMPs. Map daily PET (reference to grass) from the
C   DAY_PET0 array back to original POTET array.  Init SUM_HEAD array if
C   it is the first day of the month. This is where the daily time step
C   starts.
c
            CALL DAILY_VARIABLES_INIT(total_flow_g404,rfls155a ,
     $      s31dsg_to_bay,total_reuse_res,add_flow_to_sta34 ,
     $      ws_from_acmres_lwdd, ws_to_sta34_frm_lok ,
     $      water_supply_to_holyl,tot_flw_to_c9r_frm_nlkblt ,
     $      tot_lok_to_eaares,RMFLWS12,RMFLWS333,s333_reg_lec ,
     $      asr_rec_ws,asr_rec_ws_cnty,desgn_cap_canl_ws ,
     $      total_reuse_ws,temp_volwtr,volwtr,mod_delta_stor ,
     $      depth_levee_spg_max_from_gw,bmp_makeup_water,
     $      rmin_lok_stg_del_est,day_pet0,NYDAY ,tempf_volwtr,
     $      day_pet0_lok,nsvarea,ntotal_no_import_areas ,jday,
     $      lok_split_option,fract_mds_res,fract_mds_litzone ,bmp_opt,
```

```
     $      isem_flg,sem_dmnd_orig,total_et_red ,ws_to_l8_via_s352,
     $      vol_rec_from_cpbasr,cfsoth_res ,cfsoth_litz,numd,iyr,
     $      s333_reg_lec_abve_sch,s333_reg_lec_bel_sch,
     $      vol_rec_from_acmasr,runoff_to_sta,rlokreg_to_l8)
c
c  Counter for no of days in simulation.
c
c
c     For running mean of EAA runoff for FC backpumping
c
            ndays_sim = ndays_sim + 1
c
c     For running mean of EAA runoff for FC backpumping
c
            ndays_sim_sd = ndays_sim_sd + 1
c
C   GW_RCHG_UPDATE
C   Update GW recharge (RCHG) due to wellfield withdrawal & supply for
C   agricultural irrigation. Update daily GW recharge with average daily
C   historical pumpage for each month. Update RCHG with PWS well
C   pumpages and with industrial and residential self-supplied
C   withdrawals.  If LEC_ET is used then daily values of irrigation
C   supplies, total irrigation demands supplies and shortages, and daily
C   adjusted unsaturated zone ET for all LECSA cells are computed. Then
C   RCHG is updated with irrigation supplies taking into consideration
C   fraction of landscape irrigation served by PWS. Determine irrigation
C   supply from M-Canal for appropriate grid cells (BSN 41) & L8 canal
C   for other than sugar cane for appropriate cells (BSN 43).
c
            CALL GW_RCHG_UPDATE(wfld,ilkmg,imcnl,day_et_unsat,
     $      day_et_poten ,day_net_irr,day_et_unsat_irr,
     $      day_et_unsat_nonirr,add_dmndagl8,struc_name_meas,il8)
c
c   Adjust detention depth when water table encroaches the
c   unsaturated zone (h(node) > erz(node). When this occurs
c   the depth of surface water to infiltrate into ground
c   before runoff takes place is limited to bring the soil
c   moisture to its holding capacity.
c
            DO node = 1,isum(maxy+1)
              IF (lutyp(node) .ne. 0) THEN
               IF(use_lec_et * lec_node_in_model(node) .ne.  0.and.ibsn
     $         (node) .ne.  40.and. frac_irr_area_t(node) .ne. 0.) THEN
                  IF (erzd(node).eq.0.)THEN
                     whc = 0.0
                  ELSE
                     whc = smmax(node) / ( (GDAR/43560.)*(1.0
     $               -carfact(node))*erzd(node) )
                  ENDIF
                  swhcap = whc*erzd(node)
                  IF (h(node) .ge. ells(node)-erzd(node)) THEN
                     dpthw_excess = (h(node)-ells(node)+erzd(node)) * s
     $               (node)
                     deten(node) = swhcap - (solmx(node)+dpthw_excess)
                     deten(node) = amax1(deten(node),0.0)
                  ELSEIF(h(node) .ge. ells(node)-srz(lutyp(node))) THEN
                     deten(node) = swhcap - solmx(node)
```

```
                 ELSE
                    deten(node) = deten_def(lutyp(node))
                 ENDIF
                 deten_c(node) = deten_def(lutyp(node))
              ELSE
                 deten(node) = deten_def(lutyp(node))
                 deten_c(node) = deten_defc(lutyp(node))
              ENDIF
            ELSE
              deten(node) = 0.0
            ENDIF
          END DO
          npts=0
C   OFM_CANL
C   For canals with unique sw coeffs NOTE: ofm_canl maps all canals
C   with nodcr & 4 lutyp against ofmc_canl with icanal_swint_dep and
C   4 lutyp
C
          DO  n = 1,nch
            DO ic=1,nodcr(n)
              npts=npts+1
              nodec(ic)=nodecn(npts)/10
              if (icanal_swint_dep(n) .eq. 0) THEN
                 deten_cnl(n,ic) = deten_c(nodec(ic))
                 ofm_canl(n,ic,1) = ofmc(lutyp(nodec(ic)),1)
                 ofm_canl(n,ic,2) = ofmc(lutyp(nodec(ic)),2)
                 ofm_canl(n,ic,3) = ofmc(lutyp(nodec(ic)),3)
                 ofm_canl(n,ic,4) = ofmc(lutyp(nodec(ic)),4)
              endif
            ENDDO
          ENDDO
c
c
c  Levee seepage from Everglades/Reservoir(s) into canal system.
c
c         nodebf = 23-minx(44)+1+isum(44)
c         nodeag = 23-minx(48)+1+isum(48)
c         print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $    ,'  BEF LVSEEP'
c         print *,solmx(nodebf),pond(nodebf),ells(nodebf)
c    $     ,'   BEF LVSEEP2'
c         print *,h(469),pond(469),rchg(469),'  BEF LVSEEP'
C
C   LVSEEP
C   Levee seepage from Everglades/reservoir(s) into canal system.
C   Calculates seepage beneath protective levees into borrow canals.
C   For each reservoir cell calculate seepage at maximum depth for
C   determination of available storage. Also make sure that seepage rate
C   does not result in reverse of gradient. Then calculate volume limit
C   for seepage during each time step.  STEEPCFS & DPTH_SEEP_RES is
C   removed from VOLWTR.
C
          CALL LVSEEP(IDYMAX,lvseep_to_l33 ,lvseep_to_c6,prevqu,rain,
     $    crmin,volwtr,kflo0 ,jday,tseep_divers,
     $    tempf_volwtr ,tempf_weircff,depth_levee_spg_max_from_gw ,
     $    sumrain_unadj,rlvseep_frm_cnl,total_small_afact,node_wd,
     $    ilvflw_for_slope,HWSEEP_F_L30,HDSEEP_F_L30,ilv_f_l30,
```

```
     $      no_of_l30spnod,icol_f_l30,irow_f_l30,ipos_lv_l30,ltotal,
     $      daily_seep,seepmon)
c           print *,h(469),pond(469),rchg(469),'  AFT LVSEEP'
c----------------------
c     Input weekly s65e flows and rainfall excess for LOK for
c     climate operations
c
      IF (itrib_hydro_cond_opt .EQ. 1) then
         iday_of_week = idofw(month,iday,ifyr+iyear-1)
         IF (iday_of_week .eq. 6 ) THEN
            READ (134,'(10x,f11.2,f10.0)') trib_rf_et ,
     $      s65e_runff_wkly
         ENDIF
      ENDIF

c   ETCOMP
C   Map daily rainfall values from day_rain array to rain array and
c   update ponding with today's rainfall.
C   Note: monthly totals are computed earlier. See code after loop 30.
C   Then - ET calculation for each cell.  Computes ET components,
C   infiltration, ponding, recharge, etc.  ET is calculated based on
C   adjustment coefficients for land use types 7/8/9. Then KC is
C   calculated based on ETK; adjust coefficients; ET_CAL_ADJ_FACT. Then
C   GWD [distance from land surface to water table] is calculated and
C   used to determine the type of ET & ET_in_res calculations. Used to
C   update ETM & RES_ET for each node and then calculate ponding,
C   volwtr, rain, res_rain, etc.
C
            DO 20 I = MAXY,1,-1
               DO 10 j = ISUM(I) + 1,ISUM(I + 1)
                  rain(j) = day_rain(j,iday)*rf_factor(month)
                  IF (lutyp(j) .eq. 14) THEN
                     rain_mm = rain(j)*25.4
                     IF (rain_mm .gt. 1.2) THEN
                        effect_rain_mm = -1. 013+ 0. 824* rain_mm +
     $                  0. 00075* rain_mm ** 2.0
                        effect_rain = effect_rain_mm / 25.4
                        effect_rain = amin1(effect_rain,rain(j))
                     ELSE
                        effect_rain = 0.0
                     ENDIF
                  ELSE
                     effect_rain = rain(j)
                  ENDIF
                  rain(j) = effect_rain
                  IF((day1).and.(iday.eq.1))THEN
                     continue
                  ELSE
                     rfm(j) = rfm(j) + rain(j)
                  ENDIF
                  DO ir = 1,noresincell(j)
                     ia = ires_index_in_cell(j,ir)
                      IF(sfactor(ia).LT.sfactmin.and.ires_small_sim
     $                (ia) .eq. 'YES') THEN
c                       if (resname(ia) .eq.'L8BPRS')
c    $                  print *,resname(ia),stage_res(j,ir)
c    $                  ,pond_in_res(j,ir),' BEF RAIN'
```

```
c
                        volwtr(ia) = volwtr(ia) - rain(j)/12.*sfactor
     $                  (ia) *gdar/86400.
                        IF (stage_res(j,ir) .gt.ells_in_res(j,ir)) THEN
                           pond_in_res(j,ir) = pond_in_res(j,ir)
     $                     + rain(j)/12.
                           stage_res(j,ir) = ells_in_res(j,ir) +
     $                     pond_in_res(j,ir)
                        ELSE
                           dpth_bel_ls_res = (ells_in_res(j,ir)
     $                     - stage_res(j,ir))*s(j)
                           pond_in_res(j,ir) = amax1(rain(j)/12.
     $                     - dpth_bel_ls_res,0.0)
                           stage_res(j,ir) = stage_res(j,ir)
     $                     + amin1(dpth_bel_ls_res,rain(j)/12.)
     $                     /s(j) + pond_in_res(j,ir)
                        ENDIF
                        res_rain(ia) = res_rain(ia) + rain(j)/12.
     $                  *gdar*sfactor(ia)/43560.
c                       if (resname(ia) .eq.'L8BPRS')
c                       print *,resname(ia),stage_res(j,ir),'  RAIN  '
c    $                  ,ells_in_res(j,ir),pond_in_res(j,ir)
                        call etcomp(-1,ET,J,LUTYP_IN_RES(J,ir),JDAY ,
     $                  depth_of_deficit,stage_res,ET_outside_res,
     $                  ia,ir,sfactor(ia),ells_in_res,0)
                        volwtr(ia) = volwtr(ia) + ET *sfactor(ia)
     $                  *gdar/86400.
                        ETM(J) = ETM(J) + ET * sfactor(ia)
                        res_et(ia) = res_et(ia) + ET * gdar *
     $                  sfactor(ia) / 43560.
                     ELSE
                       IF (resname(ia) .eq. 'STA1E ') THEN
                          res_rain(ia) = res_rain(ia) + rain(j)/12.
     $                    *gdar*sfactor(ia)/43560.
                          call etcomp(1,ET,J,LUTYP_IN_RES(J,ir),JDAY ,
     $                    depth_of_deficit,stage_res,ET_outside_res,
     $                    ia,ir,sfactor(ia),ells_in_res,0)
                          res_et(ia) = res_et(ia) + ET * gdar *
     $                    sfactor(ia) / 43560.
                       ENDIF
                       DPTHGWrn = (ells(j)-h(j)) * s(j)
                       volwtr(ia) = volwtr(ia) - rain(j)/12.*sfactor
     $                 (ia) *gdar/86400.-amin1(rain(j)/12.,DPTHGWrn)
     $                 *(1.0-sfactor(ia))*gdar/86400.
                       volwtr(ia) =volwtr(ia)+daily_seep_loss_res(j)
     $                 *gdar*sfactor(ia)/86400.
                     ENDIF
c                    if (resname(ia).eq.'C9RES ') print *,volwtr(ia)
c    $               ,rain(j)/12.,ET,res_et(ia),'   OOOOO'
                  END DO
                  POND(j) = POND(j) + rain(j)/12.
   10          CONTINUE
   20       CONTINUE
c           print *,volwtr(11),' AFT RAIN AND ET'
c
c
C   INDX_LOK_DEL
```

```
C   Compute index for condition for LOK delivery to major EAA basins in
C   [MAIN.F] prior to AVAIL_RES_STOR_ADJSUT.F.  It computes index
C   internally for condition for LOK delivery to major EAA basins. Used
C   for segmenting potential delivery.

            if (nint(total_lec_dmnd_lok) .gt. 200) then
               indx_lok_del = 1
            else
               indx_lok_del = 2
            endif
c
c   AVAIL_RES_STOR_ADJUST
C   Accumulate rainfall and ET for each cell for estimating stages in
C   above ground reservoirs {sumrain & sumrain_unadj}.  Adjust available
C   storage in reservoirs due to differences between actual reservoir
C   size and grid size.
c
            DO node = 1,isum(maxy+1)
               DO iresnum = 1,ntotres
                 IF (sfactor(iresnum) .GE. sfactmin .OR.ires_small_sim
     $           (iresnum) .EQ. 'NO ') THEN
                   ires_indx_no = iresnum
                   DO k = 1,nnodes(iresnum)
                      IF (node .eq. node_reserv(iresnum,k)) THEN
                         CALL AVAIL_RES_STOR_ADJUST(DPTHGW_for_rain,
     $                   SUMRAIN_unadj,sumrain,rain,stage_res,
     $                   depth_levee_spg_max_from_gw,volwtr,node,
     $                   ires_indx_no)
                      ENDIF
                   ENDDO
                 ENDIF
               ENDDO
            ENDDO
C
c   Determine runoff/supplemental demands in EAA basins
c
C
C   AGAREA
C   Simulates water table management practices in the EAA by forcing the
C   water table to reside within a very narrow range. Excess water is
C   removed from the soil and routed through the outlet structures or to
C   a reservoir. Irrigation requirements are met from a source (LOK or
C   reservoir) and added to the recharge term. Note: total_eaa_dmnd
C   includes MIAMI + NNR_HILL + WPB  basins + SUGAR RANCH. Whenever
C   there is runoff, demand is solely met by rainfall {dmnd_met_by_rain}
C   since rainfall is source of runoff.  Calculate demand for S-4 &
C   S-236 basins & major EAA basins. Add supplemental volume required to
C   meet crop requirements to soil moistureduring days when deliveries
C   from LOK are to be made. Calculate availablestorage in EAA for
C   excess supply of water from LOK when SSM is in effect.
C   NOTE: crop coefficients [KC] are calculated again.
C   NOTE: For IBSN=7, pond calculation based on if LU.NE.'WETLAND'.
C   ELSEIF G88 & ISTAOPT(6)=0.
C   Determines portion of demand in EAA met by rainfall and/or met by
C   local storage when no RO is occurring.  Route runoff from
C   agricultural area in L-8 basin to L-8.  Calculate supplementary
C   demand for EAA basin.
```

```
c
c           print *,h(nodebf),pond(nodebf),ells(nodebf),solmx(nodebf)
c    $      ,'  BEF AGAREA'
            CALL AGAREA(JDAY,SOLMDPH,kflo0,DMDS4,DMDS236,RAIN,
     $      agrunoff_to_l8,dmnd_met_by_rain,dmnd_met_by_local_storage,
     $      total_eaa_dmnd,total_et_red,agrunoff_to_l8w,agrunoff_to_l8s,
     $      isem_flg,dmnd_sugh_orig,SUPP_EAA_DMND,sum_eaa_dmnd_ssm,
     $      alocnnr_potdel,pot_del_to_eaa_basin,sum_alocnnr,
     $      indx_lok_del,total_small_afact, dmndro, itypern,
     $      struc_name_meas,total_area_298d_grid)


C
c   Determine water supply needs to maintain canals in LEC.
c
c           print *,'  BEF WSNEEDS'
c            call match('C4   ',1,5,cnm,nch,5,imatch_c4)
c            print *,hdc(imatch_c4,iseas),'  BEF WSNEEDS ',iseas
C
C   WSNEEDS
C   Determines total surface water requirement for water supply in LEC.
C   The need is defined as the volume of water required to maintain the
C   canals at desired minimum levels during dry periods. Water, if
C   available,is delivered from the appropriate WCA or LOK.  The needs
C   are determined for every canal reach to be maintained by the
C   storage areas.  Routine sets minimum level in canals which can be
C   dependent on WCA stages; time dependent floor elevation for WCAs;
C   set condition for change in min canal level from normal ops; checks
C   for periods of heavy rainfall.  NOTE: ICMB: Canal number for canal
C   of interest along main branch of canal network. Initially is the
C   most downstream reach in canal network which is input in MODEL_
C   DEFINITION_INFO file.  ICTFST & ICTSCND are canal numbers along
C   level one and two tributary. Calculates volume of water required
C   to maintain canal ICTSCND along level two tributary or canal ICTFST
C   in level one tributary. Check if # for upstream canal reach = canal
C   in main branch ICMB. If equal, the total water supply req. for level
C   one tributary has been determined, thus continue along the main
C   branch. If not equal, total required for level one tributary have
C   yet to be determined.
C
            CALL WSNEEDS(NBRANCH,CRMIN0,JDAY,PREVQU,rain , volwtr,
     $      agrunoff_to_l8w,nsvarea,tempf_volwtr , rloc_seep_for_ws,
     $      tempf_weircff,rloc_excess_for_ws ,
     $      rloc_excss_to_canl_next_step,rlv_seep_to_canl_c11b,
     $      CRMIN_bel_flr,regstg_wca,idn_wca_indx,n_intcnl,
     $      ifloor_line_indx,iconvey_canal_no_wca,trig_wca_stg,
     $      crmin,month_beg_min_mod_flr,month_end_min_mod_flr,
     $      iday_beg_min_mod_flr,iday_end_min_mod_flr,numd,
     $      ivary_slope_daily_opt,add_dmndagl8)
c           print *,'  AFT WSNEEDS'
c
c   LOCWSLWDD
C   Deliveries of excess water into LWDD from C-51,E-4,and
c   Hillsboro canals for water supply purposes.  Determines water supply
C   deliveries into LWDD from local sources of water such as C-51 basin
C   andexcess water from LWDD canals. Determines structure flows as a
C   functionof GW level and/or canal stage; apply water supply, FC, and
```

```
C   environmental rules.  First meet the water supply needs from within
C   the LWDD with excess water if any (WSNDE1 < 0.0; WSND2 < 0.0; WSDES
C   < 0.0 sections).  Calculate available water supply and need to be
C   met by C-51; Hillsboro.
c
            ihlwdd = 0
            do kk = 1,nflpts
               local_lwdd_struc_name = struc_name_meas(kk)
               if (local_lwdd_struc_name(1:2) .eq. 'CS') then
                   ihlwdd = ihlwdd + 1
               endif
            enddo
            if (ihlwdd .ne. 0) istr_count = nflpts
            if (ihlwdd .eq. 0)
     $      CALL LOCWSLWDD(WSNDE1,WSNDE2,WSNDE3,WSNDES,WSNDESE ,CRMIN,
     $      VOLWTR,PREVQU,RAIN,istr_count ,rlocal_ws_to_e1,
     $      rlocal_ws_to_e2,rlocal_ws_to_e3 ,rlocal_ws_to_es,
     $      rlocal_ws_to_ese)
c
            stagelo_beg = stagelo
C   OPT_PRIOR_USE_ASR_FLEX_CAL - implement flexibility in priority
C   RES/ASR & LOK in meeting demands in Caloosahatchee and St Lucie
C   basins.  CAL_ASR_FLAG - flag for Caloosahatchee basin/reservoir
C   (1 - ASR exists, 0 - no ASR exists)
C   IUSE1_ASR_FIRST - Based on opt_prior_use_asr_flex_cal; stagelo <
C   stgref_asr_rec ; cal_asr_flag. It is used with c43dmd < 0 or
C   iuse_asr1 = 0 to launch SSM_CUTBACK.F. For Caloosahatchee, determine
C   allocation of water for LOKSA first.  It is also commented out in
C   CALOOS.F where it is used  to evaluate allocation for C43.
C
            if (opt_prior_use_asr_flex_cal .eq. 'YES') then
              if (stagelo .lt. stgref_asr_rec .and. cal_asr_flag
     &         .eq. 1 ) then
                iuse1_asr_first = 1
              else
                iuse1_asr_first = 0
              endif
            else
              iuse1_asr_first = 1
            endif
c
c   ASR
C   Meet EAA basin demands with excess water from C51 ASR after taking
c   into account demands in C51/LWDD. Only use C51 ASR for this
C   purpose IF appropriately input in asrinput.dat file AND no ASR
C   systems are proposed within EAA basins.
c
            flow_from_asr_for_ws_eaa = 0.0
            DO ib = 1,neaabsn-1
               IF (nasr_to_eaa(ib) .gt.  0.and. iasr_opt_eaa(ib) .eq. 0)
     $         THEN
                  vol_in = pot_del_to_eaa_basin(ib,indx_lok_del)
                  DO iar = 1,nasr_to_eaa(ib)
                     DO idmd = 1,ncnl_dest_from_asr(asrnum(3,ib,iar,2))
                        tot_cnl_dmd_asr = tot_cnl_dmd_asr + dqu
     $                  (cnl_dest_num(asrnum(3,ib,iar,2),idmd))
                        IF (cnm(cnl_dest_num(asrnum(3,ib,iar,2),idmd))
```

```
     $                  .eq. 'C51  ') THEN
                           tot_cnl_dmd_asr = tot_cnl_dmd_asr + wsnde1 +
     $                     wsnde2 + wsnde3
                        ENDIF
                     END DO
                     call asr (asrnum(3,ib,iar,2),2,vol_in ,bal_vol,1,
     $               tot_cnl_dmd_asr)
                     vol_in = bal_vol
                     tot_cnl_dmd_asr = 0.0
                  END DO
                  flow_from_asr_for_ws_eaa = amax1(pot_del_to_eaa_basin
     $            (ib,indx_lok_del),0.0)*1.9835 - bal_vol
     $            + flow_from_asr_for_ws_eaa
                  IF (pot_del_to_eaa_basin(ib,indx_lok_del) .gt. 0.0)
     $            THEN
                     pot_del_to_eaa_basin(ib,indx_lok_del) = bal_vol
     $               * 43560./1.9835
                     IF (ib .eq. 2) THEN
                        IF (alocnnr_potdel(indx_lok_del) .gt. 0.0) THEN
                           alocnnr_potdel(indx_lok_del) = (dmnd_nnr_orig
     $                     /dmnd_nnrhil_orig) * pot_del_to_eaa_basin(ib
     $                     ,indx_lok_del)
                        ENDIF
                     ENDIF
                  ENDIF
               ENDIF
            END DO
c
c   Meet EAA basin demands with excess water stored in reservoir/ASR
c   systems within EAA  IF LOK is sufficiently low.
c
            IF (runmode .eq. 'SIMUL') THEN
c   Water supply to meet EAA demands from EAA reservoir(s)
               DO irs = 1,nres_ws
                  DO js = 1,noutlets(irs)
                     IF (idshg_opt(ires_no(irs),js) .eq.  1.and.
     $               (stagelo .lt. stgref_asr_rec .or. iasr_opt_eaa
     $               (ieaa_bsn(irs,js)) .eq. 0)) THEN
C   WS_FROM_RES
C   Computes water supply discharges from reservoir to appropriate
C   destinationsto meet EAA demand from local facilities.  Water supply
C   to meet EAA demands from EAA reservoirs. Called if ASRs associated
C   with EAA exist and no runoff from ASR wells. Also if discharge
C   routing option = 1 [discharge to meet EAA basin demands] and stagelo
C   < stgref_asr_rec or iasr_opt_eaa = 1[RO not injected into ASR
C   wells]. Determine volume of water available for each outlet
C   structure. This is processed after calling ASR.  It is performed
C   twice.  First, to meet EAA demands with excess water stored in
C   reservoir/ASR systems within EAA if LOK is sufficiently low and then
C   to calculate output from appropriate reservoir for water supply
C   purposes in LEC service areas.  Processed after calling LAKE_REG_WCA
C   and routing excess water to reservoir/ASR system.  It is used to
C   recover water from EAA ASR wells to meet remaining EAA demands.
C
                        call ws_from_res(irs,js,ires_no(irs),ieaa_bsn
     $                  (irs,js),crmin ,sumrain,jday,isem_flg,dmnd_sugh,
     $                  dmnd_sugh_orig,capac_semws ,capac_sugh,
```

```
     $                  stgref_asr_rec,res_outflow_to_basin,
     $                  stagelo_beg ,pctmet,sem_dmnd_orig,0,
     $                  outflow_from_sta2_for_ws,wsndes ,wsndese,
     $                  runoff_to_sta,volwtr,frac_thru_s8_sta34 ,wsnde1,
     $                  wsnde2,wsnde3,res_outflow_env,
     $                  rloc_excess_for_ws ,tot_flw_to_c9r_frm_nlkblt,
     $                  ws_from_acmres_lwdd,issm_cutb_opt_bcyp,
     $                  pot_del_to_eaa_basin,alocnnr_potdel,
     $                  total_lec_dmnd_lok,iroute_to_wca3a_frm_holy,
     $                  iopt_env_eaar_thru_g372)
                     ENDIF
                  END DO
               END DO
               IF (stagelo .lt. stgref_asr_rec) THEN
c  Recovery from ASR within EAA basins
                  DO ib = 1,neaabsn-1
                     IF (iasr_opt_eaa(ib) .eq. 1) THEN
                        vol_in = pot_del_to_eaa_basin(ib,indx_lok_del)
     $                  *1.9835
                        DO iar = 1,nasr_to_eaa(ib)
                           call asr (asrnum(3,ib,iar,2),2,vol_in ,
     $                     bal_vol,1,tot_cnl_dmd_asr)
                           vol_in = bal_vol
                        END DO
                        flow_from_asr_for_ws_eaa = pot_del_to_eaa_basin
     $                  (ib,indx_lok_del)*1.9835 - bal_vol
     $                  + flow_from_asr_for_ws_eaa
                        IF (pot_del_to_eaa_basin(ib,indx_lok_del) .gt.
     $                  0.0) THEN
                           pot_del_to_eaa_basin(ib,indx_lok_del)=bal_vol
     $                     *43560./1.9835
                           IF (ib .eq. 2) THEN
                              IF (alocnnr_potdel(indx_lok_del) .gt. 0.0)
     $                        THEN
                                 alocnnr_potdel(indx_lok_del) =
     $                           (dmnd_nnr_orig /dmnd_nnrhil_orig)
     $                           *pot_del_to_eaa_basin(ib,indx_lok_del)
                              ENDIF
                           ENDIF
                        ENDIF
                     ENDIF
                  END DO
               ENDIF
               dmnd_nnr_aft_resasr = amax1(alocnnr_potdel(indx_lok_del)
     $         ,0.0)
            ENDIF

            IF (runmode .eq. 'SIMUL') THEN
c
c   Output from appropriate reservoir(s) for water supply purposes
c   in LEC Service Areas.
c
               DO irs = 1,nres_ws
                  DO js = 1,noutlets(irs)
                     is = istrctr_ws(ires_no(irs),js)
                     IF (idshg_opt(ires_no(irs),js) .ne.  1.and.
     $               idshg_opt(ires_no(irs),js) .ne. 3) THEN
```

```
                        call ws_from_res(irs,js,ires_no(irs),ieaa_bsn
     $                  (irs,js),crmin ,sumrain,jday,isem_flg,dmnd_sugh,
     $                  dmnd_sugh_orig,capac_semws ,capac_sugh,
     $                  stgref_asr_rec,res_outflow_to_basin,
     $                  stagelo_beg ,pctmet,sem_dmnd_orig,0,
     $                  outflow_from_sta2_for_ws,wsndes ,wsndese,
     $                  runoff_to_sta,volwtr,frac_thru_s8_sta34 ,wsnde1,
     $                  wsnde2,wsnde3,res_outflow_env,
     $                  rloc_excess_for_ws ,tot_flw_to_c9r_frm_nlkblt,
     $                  ws_from_acmres_lwdd,issm_cutb_opt_bcyp,
     $                  pot_del_to_eaa_basin,alocnnr_potdel,
     $                  total_lec_dmnd_lok,iroute_to_wca3a_frm_holy,
     $                  iopt_env_eaar_thru_g372)
                     ENDIF
                  END DO
               END DO
               total_flow_g404 = sta34_out_to_wca3anw +
     $         sta34_out_to_s140 + outflow_from_roten_to_west +
     $         holey_out_to_nwc + holy_out_to_s140
c
c   Use of ASR wells in maintaining canal levels in LEC Service areas.
c
C
C   Calculate water supply and regulatory discharges from LOK to WCAs.
C   This is processed after WS_FROM_RES.F.  Calculate recovery from
C   ASR wells associated with reservoirs IF [IRES_NO(NRES_WS)]
C   .NE.IEAA_RES_ASR(NEAABSN-1))VOL_IN = DQU(ICNL_NO)*1.9835.
C   Then call ASR.F.  Then recovery from ASR wells injected with excess
C   water from canal system and no reservoirs (e.g. C-51 ASR).
C   Then sums vol_in using DQU for each [NO_DS_CNL_WS_ASR(N,ICNL0] &
C   calls ASR.F.  Flow QU is summed based on DQU * RATIO_DEL_TO_NEED
C   (VOL_IN-BAL_VOL)/VOL_IN. Use ASR wells in proposed eastern central
C   PBC reservoir to meet western LWDD needs.
C   VOL_IN=WSNDE1+WSNDE2+WSNDE3_WSNDES+WSNDESE then process ASR.F. Use
C   ASR wells near C-51 & Hillsboro Canals to help meet the needs in
C   LWDD.  VOL_IN = WSNDE1+WSNDE2+WSNDE3 Then process ASR.F.  The
C   WSNDE(s) are recalculated based on RATIO_DEL_TO_NEED.
C
               CALL ASR_TO_LEC_WS(ires_no,flow_from_asr_ws,asr_rec_ws,
     $         asr_rec_ws_cnty,asr_rec_vol_wca,wsnde1,wsnde2,wsnde3,
     $         wsndes,wsndese,vol_rec_from_cpbasr,icounty_loc_indx,
     $         tot_cnl_dmd_asr,ic51,vol_rec_from_acmasr,
     $         asr_t_cell_volm)
c
               dischg_to_s8_frm_sta34_adj = dischg_to_s8_frm_sta34
               dischg_to_s7_frm_sta34_adj = dischg_to_s7_frm_sta34
               sta34_out_to_wca3ane_adj = sta34_out_to_wca3ane
               outflow_from_roten_thru_s8_pres =
     $         outflow_from_roten_thru_s8
            ENDIF
c
c
c   Estuarine demands for St Lucie and Caloosahatchee basins
c   Assumed to be cutback according to SSM
c
c
            IF (runmode .eq. 'SIMUL') THEN
```

```
c   initialize capacities
c   from structure books:
c              design Q      TW mainatenance stage
c   S308      14,800 cfs            14.5 ft NGVD
c   S80       16,900 cfs
c   s308cap0 = 14800.*1.9835 = 29355.8 ac-ft/day
c   s80cap0  = 16900.*1.9835 = 33521.2 ac-ft/day
c
               s308cap0 = 29355.8
               s80cap0 = 33521.2
c
c   St. Lucie estuary demands
c
               bsnsto = 0.0
               unmetdmd = 0.0
               s308cap = s308cap0
               s80cap = s80cap0
c
               IF (simsles .eq. 'NO ') THEN
                  sledmd = 0.0
               ELSE
c
                  sledmd = sle_dmd_day(jday)
               ENDIF
               sledmd0 = sledmd
c
c   Initialize capacities
c   from structure books:
c              design Q      TW maintenance stage
c   S77        9,300 cfs            11.1 ft NGVD
c   S79       28,900 cfs
c   s77cap0  =  9300.*1.9835 = 18446.6 ac-ft/day
c   s79cap0  = 28900.*1.9835 = 57323.2 ac-ft/day
c
               s77cap0 = 18446.6
               s79cap0 = 57323.2
c
c   Caloosahatchee estuary demands
c
               bsnsto = 0.0
               unmetdmd = 0.0
               s77cap = s77cap0
               s79cap = s79cap0
c
               IF (simcaes .eq. 'NO ') THEN
                  caedmd = 0.0
               ELSE
                  caedmd = cale_dmd_day(jday)
               ENDIF
               caedmd0 = caedmd
            ENDIF
c
C
C   CALOOSAHATCHEE AND ST LUCIE BASINS
C
c
            IF (runmode .eq. 'SIMUL') THEN
```

```
C   Estimate inflow into Caloosahatchee basin from S4 basin thru S235
c   positive flow means flow enters from s4 basin to C43 basin
                IF (stagelo .LE. 13.0) THEN
                 s235inf = 0.0
                 s4bsn_to_lok = s4runff(jday)
               ELSE
                 s235inf = min(s235_capac*1.9835,s4runff(jday)*
     $             s235_frac)
                 s4bsn_to_lok = s4runff(jday) - s235inf
               ENDIF
               dmnd = dmndro(2,2,itypern,jday)
               ro = dmndro(2,1,itypern,jday)
c
c   Adjust C43 basin demand or runoff due to s235 inflow
c
               dmnd_orig = dmndro(2,2,itypern,jday)
               ro_orig = dmndro(2,1,itypern,jday)
               IF (s235inf .GT. dmnd) THEN
                 ro = ro + (s235inf - dmnd)
                 dmnd = 0.0
               ELSE
                 dmnd = dmnd - s235inf
               ENDIF

               eqs77 = 0.0
               eqs79 = 0.0
               eqs308 = 0.0
               eqs80 = 0.0
               calro = dmndro(2,1,itypern,jday) / 1.9835
               IF (es_dmnd_acc_freq .eq. 'DLY') THEN
c
c   Determine allocation of water for LOK service area defined by
c   Supply-Side Management policy and fraction of demand in LOSA
c   to be met with allocation
c
                  dmnd = dmnd * frac_cal_dmnd_met_by_del
C
C   computes which zones require pumping cutbacks.  Recomputes
C   supplemental EAA demand to be met by LOK which is subject to SSM.
C   Then calculate total supplemental demand in LOSA to be met by LOK.
C   Everglades is subject to  same cutbacks as LOSA (according to SSM).
C   Calculate supply of water from   LOK to meet demands in its service
C   area according to SSM. [USE_SSM = 'YES'] then call SSM.  Give credit
C   to LOSA during dry seasonto delay implementation of cutbacks due to
C   SSM policy.  Adjust Brightonreservation according to SSM.  Adjust
C   LOK water supply deliveries to all LOSA basins.  Adjust flows to
C   S236 & S4 basin for water supply according to SSM.
C
                  CALL SSM_CUTBACK(s235inf,
     $            dmndro,dmds236,DMNDAGL8,DMDS4,dmdsemind,
     $            chgdmdistapoga,s4runff ,chgrnfistapoga,pctmet,
     $            env_ws_cutb_ssm_opt ,frac_env_met,runmode,use_ssm,agp,
     $            month,month_end_credit ,stagelo,stagelo_res,ifyr,
     $            iyear,iday,dellec,lok_targ_level ,lok_split_option,
     $            fract_rain_litzone,wup,wup0,supply_excess,demand_s236,
     $            demand_l8,demand_caloos,demand_stlucie,demand_s4bsn,
```

```
     $            demand_semind,nbrkpt_ssm1,dmnd,ro,jday,lok_ssm_cnt,
     $            ssm_vol_cutback_thres ,tdmdeaa,neaabsn,itypern,
     $            ssmlake,dmd,iday_lok_in_ssm,ssmminfrac,
     $            lok_targ_level_wet1,lok_targ_level_wet2, s235_capac,
     $            s235_frac,issm_cutb_opt_brghton,sum_alloc_ssm,
     $            sum_losa_dmnd,sem_cyp_dmnd,ibm_initc,ifreq_init,
     $            initialize_annually_opt,day1,dryseas,nwgoalsto,
     $            mon_targ_wet,iday_targ_wet,etp,rfp,pdsi,
     $            frac_dmnd_met_wt,frac_dmnd_met_wn,nbrkpt_ssmwt,
     $            nbrkpt_ssmwn,iregjulwt,iregjulwn,regstgwt,regstgwn,
     $            chgdmdnls,chgrnfnls,chgdmdnels,chgrnfnels,
     $            istokabdmdnotmet,s4bsn_to_lok,DMDS236_ORIG,DMDS4_ORIG,
     $            istokdmdaccum,dmdistok_orig,pot_del_to_eaa_basin,
     $            indx_lok_del,dmndro(1,2,itypern,jday),add_dmndagl8,
     $            pctmet0,iphase1,pcbruleopt,pcbfrac)
c   estuarine % met based on SSM, for now
                  cbslest=pctmet
                  cbc44=pctmet
c   estuarine % met based on SSM, for now
                  cbcaest=pctmet
                  cbc43=pctmet
c   DO daily checks on estuarine demands
                  stlro = dmndro(1,1,itypern,jday) / 1.9835
c
c   STLUCIE1
C   Execute St. Lucie module
C   Only performed in simulation mode.  Determines interaction (in/out)
C   flow between LOK, St Lucie Basin & St. Lucie Estuary.  Requires
C   daily timeseries of basin demand/runoff and daily time series of
C   estuarine demands. Determines structure flows as a function of GW
C   level and/orcanal stage; apply water supply, flood control and
C   environmental rules.  Calculate St Lucie estuarine demand-
C   allocation-excessive runoff 44basin demand accounting-LOK stage
C   low enough/backflow-s77 capacity exceeded/tide. Update estimated
C   flow structures: EQS308; EQS80; S308CAP; S80CAP. The
C   C44DMD_MET_BY_LOK is based on discharge.
C
                  call stlucie1(ifyr+iyear-1,month,iday,dmndro(1,2,
     $            itypern,jday) ,dmndro(1,1,itypern,jday),stltrro(jday),
     $            sledmd,cbslest ,cbc44,stagelo,bsnsto,unmetdmd,
     $            dmd_met_by_lok ,stgref,rmulti_seas_pred,
     $            clim_threshold_est)
c
c   Set the lok2stluc_est & lok2stluc_irrig variables for writing out
c   to str2x2.dss file
                  lok2stluc_est = eallok1/1.9835
                  lok2stluc_irrig = dmd_met_by_lok/1.9835
c
c   CALOOS1
C   Execute Caloosahatchee module
C   Only performed in simulation mode.  Determines interaction (in/out)
C   flow between LOK, Caloosahatchee Basin & Estuary.  Requires daily
C   time series of basin demand/runoff and daily time series of
C   estuarinedemands. Determines structure flows as a function of GW
C   level and/orcanal stage; apply water supply, flood control and
C   environmental rules.  The C43DMD_MET_BY_LOK is based on discharge.
C
```

```
                  call caloos1(ifyr+iyear-1,month,iday,dmnd,ro,caedmd,
     $            cbcaest ,cbc43,stagelo,bsnsto,unmetdmd,dmd_met_by_lok,
     $            eallok1 ,stgref,rmulti_seas_pred,clim_threshold_est,
     $            n_fc_zones,caldmndro,jday,dmnd_orig,ro_orig)
c   Set the lok2caloos_est & lok2caloos_irrig variables for writing out
c   to str2x2.dss file
                  lok2caloos_est = eallok1/1.9835
                  lok2caloos_irrig = dmd_met_by_lok/1.9835
c
               ELSE
c
c   Determine allocation of water for LOK service area defined by
c   Supply-Side Management policy and fraction of demand in LOSA
c   to be met with allocation
c
c   DO monthly checks on estuarine demands
                  IF (iday.eq.1) THEN
c          read(81,'(10x,4f12.0)') lok_supp_c43est,bflo_cap0_c43est,
c     &                            lok_supp_c44est,bflo_cap0_c44est
                     IF (simcaes .eq. 'NO ') THEN
                        lok_supp_c43est(month) = 0.0
                        bflo_cap0_c43est(month) = 999999.
                     ENDIF
                     IF (simsles .eq. 'NO ') THEN
                        lok_supp_c44est(month) = 0.0
                        bflo_cap0_c44est(month) = 999999.
                     ENDIF
c
                     lok_supp_day_c43est = lok_supp_c43est(month)/numd
     $               (month)
                     bflo_cap_c43est = bflo_cap0_c43est(month)
c
                     lok_supp_day_c44est = lok_supp_c44est(month)/numd
     $               (month)
                     bflo_cap_c44est = bflo_cap0_c44est(month)
                  ENDIF
c
c   For St Lucie Estuary only
                  stlro = dmndro(1,1,itypern,jday) / 1.9835
                  iuse_res2 = idbsnopt2
                  iuse_asr2 = stl_asr_flag
                  dmnd2_orig = dmndro(1,2,itypern,jday)
                  ro2_orig = dmndro(1,1,itypern,jday)
                  dmnd2 = dmndro(1,2,itypern,jday)
                  ro2 = dmndro(1,1,itypern,jday)
                  supp_day_c44est_tmp = lok_supp_day_c44est
                  ares02 = 0.
                  IF (iuse_res2.eq.1) THEN
                     in_out_asr2 = 0
                     qin_max02 = 99999.
                     qout_max02 = 99999.
                     ares02 = stlucie_res_area
                     rmax_dpth02 = stlucie_res_max_dpth
                     IF (iuse_asr2 .eq. 1) THEN
                        injection_max02 = asr_incap(stl_asrnum)*1.547
                        withdrawal_max02 = asr_outcap(stl_asrnum)*1.547
                        util = asreffic(stl_asrnum)
```

```
                     ELSE
                        injection_max02 = 0.0
                        withdrawal_max02 = 0.0
                        util = 1.0
                     ENDIF
c
                     bsn2est_day = bsn2est2(month,iday)
C
C   RESASR_SIM
C   Update water level in reservoirs based on RF-ET-seepage.  Then
C   calculate ET and seepage to bring HRES to zero. SPIL_OVER goes to
C   estuary not to basin.  Then all reservoir SPIL_OVER will be used
C   to meet part of remaining supplemental estuarine requirement. Do
C   ASR injection if wateris available from the reservoir and ASR well
C   not yet used to supplement reservoir in meeting estuarine demand.
C   Assumption: Reservoir releasedoes not exceed remaining runoff for
C   the day. Note model uses decay rate = 0.0 [which calculates
C   ASR_VOL & ASR_STORAGE_LOSS]; res_seepage = 0.0114 [which calculates
C   HRES].  This is not used forES_DMND_ACC_FREQ='DLY'-> daily
C   accounting.
C
                     call resasr_sim( 2, ifyr+iyear-1, month, iday, numd
     $               (month) , iuse_asr2, iuse_asr_env2, bsn2est_day ,
     $               dmnd2, ro2, supp_day_c44est_tmp, bflo_cap_c44est ,
     $               c44rf(jday) , c44pet(jday) , ares02, 0.0,
     $               rmax_dpth02, hres02, asr_vol02, in_out_asr2 ,
     $               qin_max02, qout_max02, injection_max02,
     $               withdrawal_max02 , res_asr_outflow,stagelo,util,
     $               rmin_lok_stg_del_est , opt_bsn_prio_uncond,
     $               lok_split_option,stagelo_res , n_fc_zones)
                     eqs80 = eqs80 + res_asr_outflow
                     eqs80 = eqs80 + bsn2est_day
                     IF (iuse_asr2 .eq. 1) asrvol(stl_asrnum) =
     $               asr_vol02
c
c   supp_day_c44est_tmp = revised requirement from today to end of month
c   calculate portion of original estuarine supplemental requirement for
c   the day that cannot be met by reservoir/ASR system and has to come
c   from Lake Okeechobee for today only:
                     supp_day_c44est_tmp = lok_supp_day_c44est -
     $               res_asr_outflow
c   consider (im)precision of calculations
                     IF(supp_day_c44est_tmp .lt. 0.0001) THEN
                        supp_day_c44est_tmp = 0.0
                     ENDIF
c        NOW: supp_day_c44est_tmp = remaining requirement for today only
c
c   storage volume = 48,000 ac-ft ==> 6000.0 acres @ 8 ft depth
c   hres02 <--- beginning of day reservoir water level
c   asr_vol02 <--- beginnning of day asr bubble volume
c   in_out_asr2 <--- 0:ASR not yet used, 1:ASR injection, 2:ASR recovery
c   qin_max02 = qout_max02 <--- beginning of day reservoir pump cap
c   3400 cfs  1700 cfs  (use 99999. cfs for now)
c   injection_max02 = withdrawal_max02 <--- beginning of day asr pump
c   caps.  1000 MGD = 1547.22 cfs
c
                  ENDIF
```

```
                  bflo_frac_c44est = 1.00
c
c   For Caloosahatchee Estuary only
                  IF (s235inf.gt.0.0) THEN
                     IF (dmnd_orig.gt.0.) THEN
                        addl_flow = amax1(s235inf-dmnd_orig,0.)
                     ELSE
                        addl_flow = s235inf
                     ENDIF
c   Lateral inflow from S4 basin via S235 for today could effectively
c   decrease the target supplemental LOK release or increase the maximum
c   available backflow for the remainder of the month
                     IF (lok_supp_day_c43est.gt.0.) THEN
c   addl_flow has the effect of modifying bsn2est() today
c   until the end of the month
                        rem_supp = lok_supp_day_c43est*(numd(month)
     $                  -iday+1)
                        IF (addl_flow.gt.rem_supp) THEN
c   no supplemental LOK release is needed for the rest of the month,
c   and backflow for the remainder of the month is now possible
                           bsn2est1(month,iday)=bsn2est1(month,iday)
     $                     +rem_supp
                           lok_supp_day_c43est = 0.
                           bflo_cap_c43est = addl_flow - rem_supp
                        ELSE
c   only a reduction in supplemental LOK release can be realized
                           bsn2est1(month,iday)=bsn2est1(month,iday)
     $                     +addl_flow
                           lok_supp_day_c43est = lok_supp_day_c43est
     $                     - (addl_flow/(numd(month)-iday+1))
                        ENDIF
                     ELSE
c   addl_flow has no effect on bsn2est()
c               IF (bflo_cap_c43est.eq.0.) THEN
c                  WRITE(*,*)
c     &             'lok_supp_day & bflo_cap for c43 are both zeroes!'
c                  WRITE(*,'(3i5)')ifyr+iyear-1,month,iday
c                  stop
c               ENDIF
c   increase potential ( maximum available) backflow for rest of month
                        bflo_cap_c43est = bflo_cap_c43est + addl_flow
                     ENDIF
                  ENDIF
                  iuse_res1 = idbsnopt1
                  iuse_asr1 = cal_asr_flag
                  supp_day_c43est_tmp = lok_supp_day_c43est
                  ares01 = 0.0
                  c43dmd_met_by_lok = 0.0
                  c43dmd = dmnd
c                 IF (c43dmd.le.0. .or. iuse_asr1 .eq. 0) THEN
c                  CALL SSM_CUTBACK(s235inf,
c     $            dmndro,dmds236,DMNDAGL8,DMDS4,dmdsemind,
c     $            chgdmdistapoga,s4runff ,chgrnfistapoga,pctmet,
c     $            env_ws_cutb_ssm_opt,frac_env_met,runmode,use_ssm,agp,
c     $            month,month_end_credit ,stagelo,stagelo_res,ifyr,
c     $            iyear,iday,dellec,lok_targ_level ,lok_split_option,
c     $            fract_rain_litzone,wup,wup0,supply_excess,demand_s236
```

```
c     $            demand_l8,demand_caloos,demand_stlucie,demand_s4bsn,
c     $            demand_semind,nbrkpt_ssm1,dmnd,ro,jday,lok_ssm_cnt,
c     $            ssm_vol_cutback_thres ,tdmdeaa,neaabsn,itypern,
c     $            ssmlake,dmd,iday_lok_in_ssm,ssmminfrac,
c     $            lok_targ_level_wet1,lok_targ_level_wet2, s235_capac,
c     $            s235_frac,issm_cutb_opt_brghton,sum_alloc_ssm,
c     $            sum_losa_dmnd,sem_cyp_dmnd,ibm_initc,ifreq_init,
c     $            initialize_annually_opt,day1,dryseas,nwgoalsto,
c     $            mon_targ_wet,iday_targ_wet,etp,rfp,pdsi,
c     $            frac_dmnd_met_wt,frac_dmnd_met_wn,nbrkpt_ssmwt,
c     $            nbrkpt_ssmwn,iregjulwt,iregjulwn,regstgwt,regstgwn,
c     $            chgdmdnls,chgrnfnls,chgdmdnels,chgrnfnels,
c     $            istokabdmdnotmet,s4bsn_to_lok,DMDS236_ORIG,DMDS4_ORIG
c     $            istokdmdaccum,dmdistok_orig,pot_del_to_eaa_basin,
c     $            indx_lok_del,dmndro(1,2,itypern,jday),add_dmndagl8)
c     $            pctmet0,iphase1,pcbruleopt,pcbfrac)
c                 ENDIF

c   For Caloos call to LOK first
                 IF (iuse1_asr_first.eq.0.and.c43dmd.gt.0. .and.
     $           iuse_asr1 .eq. 1) THEN
c
c   Determine allocation of water for LOK service area defined by
c   Supply-Side Management policy and fraction of demand in LOSA
c   to be met with allocation
c
                 dmnd = dmnd * frac_cal_dmnd_met_by_del
C
C   Computes which zones require pumping cutbacks.  Recomputes
C   supplemental EAA demand to be met by LOK which is subject to SSM.
C   Then calculate total supplemental demand in LOSA to be met by LOK.
C   Everglades is subject to same cutbacks as LOSA (according to SSM).
C   Calculate supply of water from   LOK to meet demands in its service
C   area according to SSM. [USE_SSM = 'YES'] then call SSM.  Give credit
C   to LOSA during dry season to delay implementation of cutbacks due to
C   SSM policy.  Adjust Brighton reservation according to SSM.  Adjust
C   LOK water supply deliveries to all LOSA basins.  Adjust flows to
C   S236 & S4 basin for water supply according to SSM.
C
                 CALL SSM_CUTBACK(s235inf,
     $           dmndro,dmds236,DMNDAGL8,DMDS4,dmdsemind,
     $           chgdmdistapoga,s4runff ,chgrnfistapoga,pctmet,
     $           env_ws_cutb_ssm_opt ,frac_env_met,runmode,use_ssm,
     $           agp,month,month_end_credit ,stagelo,stagelo_res,ifyr,
     $           iyear,iday,dellec,lok_targ_level ,lok_split_option,
     $           fract_rain_litzone,wup,wup0,supply_excess,
     $           demand_s236,demand_l8,demand_caloos,demand_stlucie,
     $           demand_s4bsn,demand_semind,nbrkpt_ssm1,dmnd,ro,jday,
     $           lok_ssm_cnt,ssm_vol_cutback_thres ,tdmdeaa,neaabsn,
     $           itypern,ssmlake,dmd,iday_lok_in_ssm,ssmminfrac,
     $           lok_targ_level_wet1,lok_targ_level_wet2, s235_capac,
     $           s235_frac,issm_cutb_opt_brghton,sum_alloc_ssm,
     $           sum_losa_dmnd,sem_cyp_dmnd,ibm_initc,ifreq_init,
     $           initialize_annually_opt,day1,dryseas,nwgoalsto,
     $           mon_targ_wet,iday_targ_wet,etp,rfp,pdsi,
     $           frac_dmnd_met_wt,frac_dmnd_met_wn,nbrkpt_ssmwt,
     $           nbrkpt_ssmwn,iregjulwt,iregjulwn,regstgwt,regstgwn,
```

```
     $             chgdmdnls,chgrnfnls,chgdmdnels,chgrnfnels,
     $             istokabdmdnotmet,s4bsn_to_lok,DMDS236_ORIG,
     $             DMDS4_ORIG,istokdmdaccum,dmdistok_orig,
     $             pot_del_to_eaa_basin,indx_lok_del,dmnd2,add_dmndagl8,
     $             pctmet0,iphase1,pcbruleopt,pcbfrac)
c estuarine % met based on SSM, for now
                   cbslest=pctmet
                   cbc44=pctmet
c estuarine % met based on SSM, for now
                   cbcaest=pctmet
                   cbc43=pctmet
c
                   alc43 = cbc43*c43dmd
c entire allocation to be met by Lake Okeechobee
                   c43dmd_met_by_lok = alc43
                   dmnd = (1.-cbc43)*c43dmd
c update structure flow
                   eqs77 = eqs77 + c43dmd_met_by_lok
                   s77cap = s77cap - c43dmd_met_by_lok

                   IF (iuse_res1.eq.1) THEN
                     in_out_asr1 = 0
                     qin_max01 = s77cap
                     qout_max01 = 99999.
                     rmax_dph01 = caloos_res_max_dpth
                     IF (iuse_asr1 .eq. 1) THEN
                        injection_max01 = asr_incap(cal_asrnum)*1.547
                        withdrawal_max01 = asr_outcap(cal_asrnum)*1.547
                        ares01 = caloos_res_area_w_asr
                        util = asreffic(cal_asrnum)
                     ELSE
                        injection_max01 = 0.0
                        withdrawal_max01 = 0.0
                        ares01 = caloos_res_area
                        util = 1.0
                     ENDIF
c
                     bsn2est_day = bsn2est1(month,iday)
C
C   RESASR_SIM
C   Update water level in reservoirs based on RF-ET-seepage.  Then
C   calculate ET and seepage to bring HRES to zero. SPIL_OVER goes to
C   estuary not to basin.  Then all reservoir SPIL_OVER will be used to
C   meet part of remaining supplemental estuarine requirement. Do ASR
C   injection if water is available from the reservoir and ASR well not
C   yet used to supplement reservoir in meeting estuarine demand.
C   Assumption: Reservoir release does not exceed remaining runoff for
C   the day. Note model uses decay rate = 0.0 [which calculates ASR_VOL
C   & ASR_STORAGE_LOSS]; res_seepage = 0.0114[which calculates HRES].
C   This is not used for ES_DMND_ACC_FREQ='DLY'-> daily accounting.
C
                     call resasr_sim( 1, ifyr+iyear-1, month, iday, numd
     $               (month) , iuse_asr1, iuse_asr_env1, bsn2est_day ,
     $               dmnd, ro, supp_day_c43est_tmp, bflo_cap_c43est ,
     $               c43rf(jday) , c43pet(jday) , ares01, 0.0,
     $               rmax_dph01, hres01, asr_vol01, in_out_asr1 ,
     $               qin_max01, qout_max01, injection_max01,
```

```
     $               withdrawal_max01 , res_asr_outflow,stagelo,util,
     $               rmin_lok_stg_del_est , opt_bsn_prio_uncond,
     $               lok_split_option,stagelo_res , n_fc_zones)
                     eqs79 = eqs79 + res_asr_outflow
                     eqs79 = eqs79 + bsn2est_day
                     IF (iuse_asr1 .eq. 1) asrvol(cal_asrnum) =
     $               asr_vol01
c   supp_day_c43est_tmp = revised requirement from today to end of month
c   Calculate portion of original estuarine supplemental requirement for
c   the day that cannot be met by reservoir/ASR system and has to come
c   from Lake Okeechobee for today only:
                     supp_day_c43est_tmp = lok_supp_day_c43est -
     $               res_asr_outflow
c   Consider (im)precision of calculations
                     IF(supp_day_c43est_tmp .lt. 0.0001) THEN
                        supp_day_c43est_tmp = 0.0
                     ENDIF
c   NOW: supp_day_c43est_tmp = remaining requirement for today only
c
c   storage volume = 320,000 ac-ft ==> 40,000 acres @ 8 ft depth
c   storage volume = 450,000 ac-ft ==> 56,250 acres @ 8 ft depth
c   hres01 <--- beginning of day reservoir water level
c   asr_vol01 <--- beginnning of day asr bubble volume
c   in_out_asr1 <--- 0:ASR not yet used, 1:ASR injection, 2:ASR recovery
c   qin_max01 = qout_max01 <--- beginning of day reservoir pump caps
c   5000 cfs  1500 cfs  (use 99999. cfs for now)
c   injection_max01 = withdrawal_max01 <--- beginning of day asr pump
c   caps.  500 MGD = 773.61 cfs
c
                    ENDIF
                   ELSE
c      IF (c43dmd.gt.0. .and. iuse_asr1 .eq. 1) THEN
                    IF (iuse_res1.eq.1) THEN
                      in_out_asr1 = 0
                      qin_max01 = s77cap
                      qout_max01 = 99999.
                      rmax_dph01 = caloos_res_max_dpth
                      IF (iuse_asr1 .eq. 1) THEN
                         injection_max01 = asr_incap(cal_asrnum)*1.547
                         withdrawal_max01 = asr_outcap(cal_asrnum)*1.547
                         ares01 = caloos_res_area_w_asr
                         util = asreffic(cal_asrnum)
                      ELSE
                         injection_max01 = 0.0
                         withdrawal_max01 = 0.0
                         ares01 = caloos_res_area
                         util = 1.0
                      ENDIF
c
                      bsn2est_day = bsn2est1(month,iday)
                      call resasr_sim( 1, ifyr+iyear-1, month, iday, numd
     $               (month) , iuse_asr1, iuse_asr_env1, bsn2est_day ,
     $               dmnd, ro, supp_day_c43est_tmp, bflo_cap_c43est ,
     $               c43rf(jday) , c43pet(jday) , ares01, 0.0,
     $               rmax_dph01, hres01, asr_vol01, in_out_asr1 ,
     $               qin_max01, qout_max01, injection_max01,
     $               withdrawal_max01 , res_asr_outflow,stagelo,util,
```

```
     $               rmin_lok_stg_del_est , opt_bsn_prio_uncond,
     $               lok_split_option,stagelo_res , n_fc_zones)
                     eqs79 = eqs79 + res_asr_outflow
                     eqs79 = eqs79 + bsn2est_day
                     IF (iuse_asr1 .eq. 1) asrvol(cal_asrnum) =
     $               asr_vol01
c   supp_day_c43est_tmp = revised requirement from today to end of month
c   Calculate portion of original estuarine supplemental requirement for
c   the day that cannot be met by reservoir/asr system and has to come
c   from Lake Okeechobee for today only:
                     supp_day_c43est_tmp = lok_supp_day_c43est -
     $               res_asr_outflow
c   Consider (im)precision of calculations
                     IF(supp_day_c43est_tmp .lt. 0.0001) THEN
                        supp_day_c43est_tmp = 0.0
                     ENDIF
c   NOW: supp_day_c43est_tmp = remaining requirement for today only
c
c   storage volume = 320,000 ac-ft ==> 40,000 acres @ 8 ft depth
c   storage volume = 450,000 ac-ft ==> 56,250 acres @ 8 ft depth
c   hres01 <--- beginning of day reservoir water level
c   asr_vol01 <--- beginnning of day asr bubble volume
c   in_out_asr1 <--- 0:ASR not yet used, 1:ASR injection, 2:ASR recovery
c   qin_max01 = qout_max01 <--- beginning of day reservoir pump caps
c   5000 cfs  1500 cfs  (use 99999. cfs for now)
c   injection_max01 = withdrawal_max01 <--- beginning of day asr pump
c   caps.  500 MGD = 773.61 cfs
c
                   ENDIF
c
c   Determine allocation of water for LOK service area defined by
c   Supply-Side Management policy and fraction of demand in LOSA
c   to be met with allocation
c
                   dmnd = dmnd * frac_cal_dmnd_met_by_del
C
C   Computes which zones require pumping cutbacks.  Recomputes
C   supplemental EAA demand to be met by LOK which is subject to SSM.
C   Then calculate total supplemental demand in LOSA to be met by LOK.
C   Everglades is subject to same cutbacks as LOSA (according to SSM).
C   Calculate supply of water from LOK to meet demands in its service
C   area according to SSM.  [USE_SSM = 'YES'] then call SSM.  Give
C   credit to LOSA during dry season to delay implementation of cutbacks
C   due to SSM policy.  Adjust Brighton reservation according to SSM.
C   Adjust LOK water supply deliveries to all LOSA basins.  Adjust flows
C   to S236 & S4 basin for water supply according to SSM.
C
                   CALL SSM_CUTBACK(s235inf,
     $             dmndro,dmds236,DMNDAGL8,DMDS4,dmdsemind,
     $             chgdmdistapoga,s4runff ,chgrnfistapoga,pctmet,
     $             env_ws_cutb_ssm_opt ,frac_env_met,runmode,use_ssm,
     $             agp,month,month_end_credit ,stagelo,stagelo_res,ifyr,
     $             iyear,iday,dellec,lok_targ_level ,lok_split_option,
     $             fract_rain_litzone,wup,wup0,supply_excess,
     $             demand_s236,demand_l8,demand_caloos,demand_stlucie,
     $             demand_s4bsn,demand_semind,nbrkpt_ssm1,dmnd,ro,jday,
     $             lok_ssm_cnt,ssm_vol_cutback_thres ,tdmdeaa,neaabsn,
```

```
     $             itypern,ssmlake,dmd,iday_lok_in_ssm,ssmminfrac,
     $             lok_targ_level_wet1,lok_targ_level_wet2, s235_capac,
     $             s235_frac,issm_cutb_opt_brghton,sum_alloc_ssm,
     $             sum_losa_dmnd,sem_cyp_dmnd,ibm_initc,ifreq_init,
     $             initialize_annually_opt,day1,dryseas,nwgoalsto,
     $             mon_targ_wet,iday_targ_wet,etp,rfp,pdsi,
     $             frac_dmnd_met_wt,frac_dmnd_met_wn,nbrkpt_ssmwt,
     $             nbrkpt_ssmwn,iregjulwt,iregjulwn,regstgwt,regstgwn,
     $             chgdmdnls,chgrnfnls,chgdmdnels,chgrnfnels,
     $             istokabdmdnotmet,s4bsn_to_lok,DMDS236_ORIG,
     $             DMDS4_ORIG,istokdmdaccum,dmdistok_orig,
     $             pot_del_to_eaa_basin,indx_lok_del,dmnd2,add_dmndagl8,
     $             pctmet0,iphase1,pcbruleopt,pcbfrac)
c   Estuarine % met based on SSM, for now
                   cbslest=pctmet
                   cbc44=pctmet
c   Estuarine % met based on SSM, for now
                   cbcaest=pctmet
                   cbc43=pctmet

                  ENDIF
c
c   Execute St. Lucie module
c
                  bsn2est_day = bsn2est2(month,iday)
C   STLUCIE2
C   Only performed in simulation mode.  Determines interaction (in/out)
C   flow between LOK, St Lucie Basin & St. Lucie Estuary.  Requires
C   daily time series of basin demand/runoff and monthly daily time
C   series of estuarine demands. Determines structure flows as a
C   function of GW level and/or canal stage;  apply water supply, flood
C   control and environmental rules.  Calculate St Lucie estuarine
C   demand, but allocation is not initially performed.  Supplemental
C   releases are based on lok_supp_day > 0 [release required from LOK].
C   Also checks bflo_cap .GE. bflo_frac * eflc44 [FRACTION OF BACKFLOW
C   USED IN C44]. Excessive runoff 44 BASIN demand accounting-LOK stage
C   low enough/backflow-s77 capacity exceeded/tide. The C44DMD_MET_
C   BY_LOK is based on allocation.  LOK_SUPP_DAY > 0-> supplementary
C   releases from LOK required.
C
                  IF (iuse_lok2.eq.1) THEN
                     call stlucie2(ifyr+iyear-1,month,iday,numd(month),
     $               dmnd2, ro2, stltrro(jday),sledmd,cbc44,stagelo,
     $               supp_day_c44est_tmp, bflo_cap_c44est,
     $               bflo_frac_c44est,lok_supp_c44est(month),
     $               c44dmd_met_by_lok,bsn2est_day, stgref,
     $               rmulti_seas_pred,clim_threshold_est,cbslest,
     $               dmnd2_orig,ro2_orig)
c   Set the lok2stluc_est variables for writing out to str2x2.dss file
                     lok2stluc_est = (sledmd_met_by_lok)/1.9835
c   Remaining estuarine supplemental requirement for the day is zero
                     supp_day_c44est_tmp = 0.0
                  ELSE
                     call stlucie2(ifyr+iyear-1,month,iday,numd(month),
     $               dmnd2, ro2, stltrro(jday),sledmd,cbc44,stagelo,
     $               0.0, bflo_cap_c44est,bflo_frac_c44est,
     $               lok_supp_c44est(month), c44dmd_met_by_lok,
```

```
     $               bsn2est_day, stgref,rmulti_seas_pred,
     $               clim_threshold_est,cbslest,dmnd2_orig,ro2_orig)
c   Set the lok2stluc_est variables for writing out to str2x2.dss file
                     lok2stluc_est = 0.0
                  ENDIF
c
c   Set the lok2stluc_irrig variables for writing out to str2x2.dss file
                  lok2stluc_irrig = c44dmd_met_by_lok/1.9835

c
c execute Caloosahatchee module
c
                 bsn2est_day = bsn2est1(month,iday)
C
C   CALOOS2 is the same as CALOOS1
C   But it requires daily time series of basin demand/runoff and a
C   monthly time series of estuarine demand. Determines structure flows
C   as a function of GW level and/or canal stage; apply water supply,
C   flood control, and environmental rules.  Caloosahatchee estuarine
C   demand-allocation-excessive runoffC43 basin demand accounting -
C   LOK stage low enough/backflow S77 cap exceeded/tide.  The
C   C43DMD_MET_BY_LOK is based on allocation.
C
                  IF(iuse_lok1.eq.1) THEN
                     call caloos2(ifyr+iyear-1,month,iday,numd(month),
     $               dmnd,ro, caedmd,cbc43,stagelo,
     $               supp_day_c43est_tmp, bflo_cap_c43est,
     $               bflo_frac_c43est,lok_supp_c43est(month),
     $               c43dmd_met_by_lok,iuse1_asr_first,bsn2est_day,
     $               stgref,rmulti_seas_pred,clim_threshold_est,
     $               n_fc_zones, cbcaest,caldmndro,jday,dmnd_orig,
     $               ro_orig)
c   Set the lok2caloos_est variables for writing out to str2x2.dss file
                     lok2caloos_est = c43est_met_by_lok/1.9835
c   Remaining estuarine supplemental requirement for the day is zero
                     supp_day_c43est_tmp = 0.0
                  ELSE
                     call caloos2(ifyr+iyear-1,month,iday,numd(month),
     $               dmnd,ro, caedmd,cbc43,stagelo, 0.0,
     $               bflo_cap_c43est,bflo_frac_c43est,lok_supp_c43est
     $               (month), c43dmd_met_by_lok,iuse1_asr_first,
     $               bsn2est_day, stgref,rmulti_seas_pred,
     $               clim_threshold_est,n_fc_zones,cbcaest,caldmndro,
     $               jday,dmnd_orig,ro_orig)
c   Set the lok2caloos_est variables for writing out to str2x2.dss file
                     lok2caloos_est = 0.0
                  ENDIF
c
c   Set the lok2caloos_irrig variable for writing out to str2x2.dss file
                  lok2caloos_irrig = c43dmd_met_by_lok/1.9835
c
               ENDIF
            ENDIF
c
C   ALLOC_TO_EAA
C   Delivery of allocated water (SSM policy) in excess of current demand
C   in LOSA to EAA basins if needed. Only executed if USE_SSM = 'YES'.
```

```
C   Passes SUPPLY_EXCESS from SSM_CUTBACK. Rouine calculates total net
C   demand (AGP) and DMNDAGL8 = used to calculate total supplemental
C   demand in (LOSA) & L8 basin to be met by LOK. It takes excess
C   supply/allocation to EAA and calculates the amount of water
C   determined by supply-side  management plan that is in excess of
C   demand in LOSA. This excess is routed to the EAA to increase water
C   levels. The SOLMX(j) for each j basin is checked again LUTYP(j) =
C   'AGRICUL' and IBSN(j) = 40 or 42 to calculate SOLMX [equivalent
C   depth of soil moisture in soil column] based on ratio_supply_exces
C   and avail_stor adjusted by canal & small reservoir area.
c
            IF (use_ssm .eq. 'YES') call alloc_to_eaa (supply_excess
     $       ,total_small_afact)
c
C
C   EAA_NEUTRAL_CAPS
C   Determine structure flows and interior canal stages for EAA using
C   canal profiles for neutral condition from HEC-2 runs.  Calculate
C   conveyance capacities for the major EAA canals for the neutral (no
C   EAA runoff/no EAA demand) conditions. Compute demands in EAA to be
C   met during SSM.  Determine forward pumping (dependent on LOK stage).
C   Determine fraction of EAA demand to be met during days of delivery
C   from LOK. Calculate DSCHG_TO_EAA which is passed to EAASPILL.
C   Objective is to calculate gate opening for calculated discharge to
C   EAA through conveyance canal.  EAA conveyance calculations are based
C   on look-up tables prepared from HEc-2 runs with varying combinations
C   of discharge, upstream stage, and downstream stage.  Tables use
C   (S2, S3, S5A, S6, S7, S8) pumps; (S7, S8) gravity structures;
C   S354(HG-3), S351(HG-4); S352(HG-5).
C
            IF (runmode .eq. 'SIMUL') call eaa_neutral_caps(ifacteaa,
     $      pctmet ,dmnd_nnr_aft_resasr,res_outflow_to_basin,
     $      rlok_stg_beg_forpmp,rlok_stg_end_forpmp,sum_alloc_ssm,
     $      sum_losa_dmnd,ssmminfrac,alocnnr_potdel,indx_lok_del,
     $      pdsi,frac_dmnd_met_wn,frac_dmnd_met_wt,nbrkpt_ssmwt,
     $      nbrkpt_ssmwn,iregjulwt,iregjulwn,regstgwt,regstgwn,
     $      sum_eaa_dmnd_ssm_nc)
c
C   RAIN_DRIVEN_FLOW_STAGE_TARG
C   Setup rain driven flow and/or stage targets for EVERGLADES/WETLAND
C   AREA.  Rain Fall formula variable is either calculated or read in
C   via (55) depending on opt_ft_input variable. If 'YES'[pre_processed
C   read_in option] ENP rainfall based discharge to Shark River Slough
C   assign today's read-in value to RF formula variable. If not compute
C   flow targets for SRS in ENP. Type target is also used in conjunction
C   with type_flow_across_ttrail to determine the type of Shark River
C   Slough demand data to use.  Daily import stages for Everglades is
C   read in  if run is 'SIMUL' and wca_import_variation = 'DAILY' .OR.
C   opt_flow_to_ts - 'TSRFPLN' or opt_st_input. Routine then determines
C   regulation schedule for raindriven ops based on if (WCA_IMPORT_
C   VARIATION = 'DAILY' .OR. OPT_FLOW_TO_TS='TSRFPLN' OR. OPT_ST_INPUT)
C   then Check WCAENV ELSE target is constant. If the target is constant
C   then simulated stages are calculated using average import stage
C   values. WCA_NODAL_STAGE is calculated based on wca_import_variation.
c
            CALL RAIN_DRIVEN_FLOW_STAGE_TARG(nzone,regstg_wca,wcaenv ,
     $      ibot_zone_indx,itop_zone_indx,srsdmnd,n_intcnl ,jday,nyday,
```

```
     $      ntotal_no_import_areas,ntotal_no_import_areas_const ,iwca3a)
c
c   Compute discharges for LOK outlet structure to meet EAA demands and
c   for inlet structures to WCAs
c
c           print *,stage_res(nodebf,1),'  BEF ROUTE'
c           print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  BEF ROUTE'
C   HIST_FLOW_ADJUST
C   Adjust flows provided by database when necessary; only done during
C   calibration/verification based on AVVOL & KFLO. This is done for
C   structures; loaded kflo is loaded into strflw.
C
            if (runmode .eq. 'CALIB') call hist_flow_adjust
     $      (struc_name_meas,agrunoff_to_l8,c304_out_to_marsh
     $      ,prevqu)
C
C   EAA_FLOW_DISTRIB_CAPAC_SETUP
C   Compute discharges for LOK outlet structure to meet EAA demands and
C   for inlet structures to WCA. Determines fraction of EAA demand
C   to be met during days of delivery from LOK plus calculates total
C   runoff from EAA.  Approximate the portion of total runoff through
C   S-8 & S3, S7+S6 & S2, HG5A & S5A from historical data. If simulation
C   then determine moving average of daily runoff for calculation of
C   amount of flood control backpumping to LOK.  Then computes discharge
C   capacities, maximum allowable gat opening, LOK stage and tail water.
C
            CALL eaa_flow_distrib_capac_setup(jday,runoff_avg
     $      ,ifacteaa,rmonthly_loss_bmp,max_days_mean_bkpump
     $      ,sum_alloc_ssm,sum_losa_dmnd,ssmminfrac,igravity_s8_nosta
     $      ,frac_eaa_dmnd_met,sum_eaa_dmnd_ssm,indx_lok_del
     $      ,alocnnr_potdel,pdsi,frac_dmnd_met_wn,frac_dmnd_met_wt
     $      ,nbrkpt_ssmwt,nbrkpt_ssmwn,iregjulwt,iregjulwn
     $      ,regstgwt,regstgwn,iday_of_week,totrun,PFRUNF)
C
C   HOLEYLAND_INFLOW_MANAGMT
C   Evaluates if water is to be released to STAs based on LOK stage.
C   Also determines volume available for water supply deliveries from
C   the Holeyland reservoir and then target stage for Holeyland. Based
C   on WS__TO_HOLY_OPT - water is delivered to Holeyland to prevent
C   brush fires.  Total runoff and/or crop demand from EAA is calculated
C   based on 'direct' or 'indirect' runoff to Holeyland. Finally, net
C   volume of flow is calculated into the Holeyland reservoir.
C
            CALL holeyland_inflow_managmt(numd,water_supply_to_holyl
     $      ,rsiap,holyinq,totrun,PFRUNF,trig_dpth_to_wt,qhlws)
C
C   STRUC_LOK_IN_WCA_FLW
C   Compute discharges for LOK outlet structures to meet EAA demands and
C   for inlet structures to WCA.  Determines structure flows as a
C   function of gw level and/or canal stage, and reservoir stage; apply
C   water supply, flood control, and environmental rules.  The
C   calculations are based on ICODE option gen=all gen structure or
C   spc=special code unique to the structure.  Define the locations of
C   key gages in the WCA(s). Routine will return the KFLOW array
C   [structure flows for 21 specific structures] and IAGFLO [EAA basin
C   flow] to KNFLOWS.  EAA by definition includes Miami, NNRHILL, WPB
```

```
C   Basins, and Sugar Range, and excludes 298 Districts. Determines
C   water supply from LOK to L8 basin via WPB canal. Evaluates flows
C   from LOK to WCA via [culverts G-261; G263; G262; culverts for L-8
C   restoration; culvert 10A]. After this routine the MAIN.F routine
C   calculates TOTAL_EAA_SUPPLY_DELIV & TOTAL_LOSA_DELIV & ALOCNNR_DEL.
C
            CALL struc_lok_in_wca_flw(CRMIN,HOLYINQ,
     $      PCTMET,WSNEEDL8 ,JDAY,DIVERS,runoff_south,
     $      runoff_298_districts ,runoff_avg,
     $      water_supply_to_holyl,ifacteaa ,rmonthly_loss_bmp,runoff_s7,
     $      runoff_s150 ,istr_count,capacc10a_bak ,
     $      s140fc,ws_to_l8_via_s352,pot_del_to_eaa_basin,
     $      frac_eaa_dmnd_met,sum_eaa_dmnd_ssm,sum_alocnnr,
     $      indx_lok_del,hdc0,iday_of_week,totrun,rsiap,PFRUNF,
     $      trig_dpth_to_wt,qhlws,total_small_afact,
     $      add_dmndagl8,struc_name_meas)
C
            TDMDEAA1 = 0.0
            DO ia = 1,neaabsn-1
              TDMDEAA1 = TDMDEAA1+pot_del_to_eaa_basin(ia,indx_lok_del)
     $        *1.9835
            ENDDO
            total_eaa_supply_deliv = amax1(kflo(ifls3)
     $      -water_supply_to_holyl,0.0)+amax1(kflo(ifls2),0.0)+amax1
     $      (kflo(iflhg5)-ws_to_l8_via_s352 ,0.0)
            total_eaa_supply_deliv = total_eaa_supply_deliv*1.9835

            total_losa_deliv = total_eaa_supply_deliv +
     $        AMAX1(dmds236,0.0) + demand_l8*pctmet + c43dmd_met_by_lok
     $        + c44dmd_met_by_lok + dmds4 + chgdmdistapoga(jday) +
     $        chgdmdnls(jday) + chgdmdnels(jday) + amax1(kflo(iflagq),
     $        0.0) * 1.9835

            IF(alocnnr_potdel(indx_lok_del)*frac_eaa_dmnd_met .gt.
     $          amax1(kflo(ifls2),0.0)) THEN
              alocnnr_del = amax1(kflo(ifls2),0.0)*1.9835
            ELSE
              alocnnr_del = alocnnr_potdel(indx_lok_del)*
     $          frac_eaa_dmnd_met*1.9835
            ENDIF
            alochsbr_del = amax1(kflo(ifls2),0.0) * 1.9835 -
     $        alocnnr_del

            WRITE (66,
     $      '(3I5,F9.2,2X,I1,2X,4F9.2,2X,F6.4,2X,F9.2,2X,F6.4,I3)')
     $        iyear+ifyr-1, month, iday, dmd, iday_lok_in_ssm, ssmlake,
     $        supply_excess, amax1(dmd-ssmlake,0.0), amax1(dmd-dmd*
     $        pctmet,0.0), pctmet, total_losa_deliv,pctmet0,iphase1

            WRITE (61,'(3I5,23(F9.2,2x),3I5)')
     $        iyear+ifyr-1, month, iday,
     $        total_eaa_dmnd, dmnd_met_by_rain,
     $        dmnd_met_by_local_storage, SUPP_EAA_DMND,
     $        TDMDEAA1,
     $        pot_del_to_eaa_basin(is8bsn-6,indx_lok_del)*1.9835,
     $        pot_del_to_eaa_basin(is7bsn-6,indx_lok_del)*1.9835,
     $        pot_del_to_eaa_basin(is5absn-6,indx_lok_del)*1.9835,
```

```
     $        dmnd_sugh_orig,total_eaa_supply_deliv,amax1(kflo(ifls3)-
     $        water_supply_to_holyl,0.0)*1.9835,alocnnr_del,
     $        alochsbr_del,
     $        amax1(kflo(iflhg5)-ws_to_l8_via_s352,0.0)* 1.9835,
     $        (tot_runoff_excess(i)*1.9835,i=1,neaabsn-1),divers*1.9835,
     $        runoff_south(is8bsn-6)*1.9835,
     $        runoff_south(is7bsn-6)*1.9835,
     $        (runoff_south(is6bsn-6)+divers)*1.9835,
     $        runoff_south(is5absn-6) *1.9835,total_et_red,
     $        (LFORPUMP(i),i=1,neaabsn-1)
     $
            IF (opt_l8_prop) THEN
                agrnl8 = agrunoff_to_l8w + agrunoff_to_l8s
            ELSE
                agrnl8 = agrunoff_to_l8
            ENDIF

            WRITE (68,'(3I5,19(1X,F9.2),1X,I5,17(1X,F9.2))')
     $      iyear+ifyr-1,month,iday,dmndro(4,2,itypern,jday),
     $      s4dmndro(2,jday),DMDS4_ORIG,DMDS4,dmndro(4,1,
     $      itypern,jday),s4dmndro(1,jday),s4runff(jday),
     $      s235inf,s4bsn_to_lok,DMDS236_ORIG,AMAX1(DMDS236,0.0),
     $      -AMIN1(DMDS236,0.0),amax1(agp(4),0.0)/43560.,
     $      AMAX1(kflo(iflagq),0.0)*1.9835,runoff_298_districts*1.9835,
     $      demand_l8,demand_l8*PCTMET,agrnl8,dmdistok_orig,
     $      istokdmdaccum,istokabdmdnotmet,chgdmdistapoga(jday),
     $      chgrnfistapoga(jday),nlsdmndro(2,jday),nlsdmndro(2,jday)
     $      -dmndro(6,2,itypern,jday),chgdmdnls(jday),nlsdmndro(1,jday),
     $      nlsdmndro(1,jday)-dmndro(6,1,itypern,jday),
     $      dmndro(6,1,itypern,jday),dmndro(7,2,itypern,jday),
     $      chgdmdnels(jday),dmndro(7,1,itypern,jday)
c
            DO i = 1,neaabsn
               tot_runoff_excess(i) = 0.0
            END DO
c
            IF (runmode .eq. 'CALIB') SPREV = STG2STO(STAGELO,0)
c
c   Update lake stage for calibration mode
c
            IF (runmode .eq. 'CALIB') THEN
              LKETVOL = LOKETVOL(STAGELO,MONTH,rf_factor_lok)
              RMETVOL = RMETVOL + LKETVOL
              VOLTOLO = -1.9835*(KFLO(IFLS3)+KFLO(IFLS2)+KFLO(IFLHG5)
     $        +KFLO(IFLS77)+KFLO(IFLS308)+KFLO(IFLAGQ)+KFLO(IFLC10A)
     $        +TWSUPLY)+KFLO(IFLS65E)*1.9835+tcnsrnff(jday)
     $        -fplresws(jday)-dmdsemind(jday)-hist_lok_bndry_flw(1,jday)
     $        +hist_lok_bndry_flw(2,jday)-hist_lok_bndry_flw(3,jday)
     $        +hist_lok_bndry_flw(4,jday)-hist_lok_bndry_flw(5,jday)
     $        +hist_lok_bndry_flw(6,jday)-hist_lok_bndry_flw(7,jday)
     $        +hist_lok_bndry_flw(8,jday)-hist_lok_bndry_flw(9,jday)
     $        +hist_lok_bndry_flw(10,jday)-hist_lok_bndry_flw(11,jday)
     $        +uisrnff(jday)

              FINSTOR = SPREV - LKETVOL + CFSOTH*1. 9835+ VOLTOLO
c
c   Output components of LOK ET
```

```
c
               if (lp(10)) then
                 write(181,'(i5,2i4,3x,f6.2,4f8.0)') ifyr+iyear-1,month
     $           ,iday,stagelo,LKETVOL/1.9835
               endif

               STAGELO = STG2STO(FINSTOR,1)
               DO I = 1,NCALCPT
                  JSTR = JSTR + 1
                  STRFLW(JSTR,IDSS) = KFLO(ICALCPT(I))
               END DO
C
               DPTHGW = (ELLS(IG3A28) - H(IG3A28)) * S(IG3A28)
               IF (POND(IG3A28) .GT. DPTHGW) THEN
                  STG3A28 = POND(IG3A28) - DPTHGW + ELLS(IG3A28)
               ELSE
                  STG3A28 = (POND(IG3A28)-DPTHGW)/S(IG3A28) + ELLS
     $            (IG3A28)
               ENDIF
            ENDIF
c
c   Compute volume of demand to maintain appropriate STAs
c   at minimum levels for environmental purposes
c
            DO ia = 1,NMAWS
               IF (NMAWS .GT. 0) THEN
                  IF (sfactor(indxaws(ia)) .GE. sfactmin .or.
     &            ires_small_sim(indxaws(ia)) .eq. 'NO ') THEN
                     CALL lok_ws_to_larger_res(IA,NUMD,WSUPLY,SUMRAIN
     &               ,RAIN,JDAY,total_afact)
                  ELSE
                     CALL lok_ws_to_small_res(IA,NUMD,WSUPLY)
                  ENDIF
               ENDIF
            ENDDO
c
c   Reduce demand for appropriate canals according to SSM
c   LWDD canals(E-1,E-2,E-3,near Boca wellfield)
c
            wsnde1 = wsnde1 * pctmet
            wsnde2 = wsnde2 * pctmet
            wsnde3 = wsnde3 * pctmet
            wsndese= wsndese* pctmet
c
c   Additional canals
c
            DO kk = 1,no_canals_ssm
              DQU(icanal_ssm(kk)) = DQU(icanal_ssm(kk))*pctmet
              reduct_demand = (1.0 - pctmet)*DQU(icanal_ssm(kk))
              icnl_dns_ssm=icanal_ssm(kk)
   5          icnl_dns_ssm=iup_canal_ws(icnl_dns_ssm)
              IF (icnl_dns_ssm.le.nch) THEN
                 dqu(icnl_dns_ssm)=dqu(icnl_dns_ssm)
     $           - reduct_demand
                 dqu(icnl_dns_ssm)=amax1(dqu(icnl_dns_ssm),0.0)
                 go to 5
              ENDIF
```

```
            ENDDO
c
C   REUSENORES
C   Reuse water directly to canals or cells outside reservoirs.  Adds
C   reuse water to appropriate locations.  Reuse is sent to canals via
C   DQ; UPSTREAM_INFLOW; DQU.  Or reuse is sent to grid cell locations:
C   Calculate POND(NODE_REUSE(I,K)) if RESNAME_REUSE(I,K) = 'NORES'
C   else AVAIL_VOL_REUSE_RES & TOTAL_REUSE_RES.
c
            IF (no_of_reuse_plnts .gt. 0) CALL REUSENORES(total_reuse ,
     $      dqu,upstream_inflow,total_reuse_ws,resname_reuse,
     $      total_afact,sumrain,rmax_stage_reuse,stage_res)
c
c   Execute subroutine to determine Lake Okeechobee deliveries for water
c   supply purposes to WCAs,and LEC
c
c           print *,stage_res(nodebf,1),'  BEF LAKE_NONREG_WCA'
C
C   LAKE_NONREG_WCA
C   Determine LOK deliveries for water supply purposes to WCA and LEC.
C   Main purpose is to call FLOW_TO_STAS.F which routes water supply
C   (including makeup water) into STAs and WS_TO_RES.F which routes
C   water supply to reservoirs. Determine current water supply
C   backpumping zone for LOK and set conditions for change in water
C   supply backpumping to LOK from normal operations. Then set up
C   fraction of expected delivery to be made from WCAs.  Calculate
C   running mean of simulated lake stages to determine if lake is
C   rising/falling [used as a condition for injection of LOK water
C   into proposed ASR].  Determine if criteria are met for environmental
C   water supply releases from LOK to WCA3A. Routine routes EAA runoff
C   to ASR reservoir if there are no LEC demands and no environmental
C   demands in Everglades.  Delivers water from LOK to maintain STA34
C   via Miami Canal during dry periods. Route water from LOK to Seminole
C   and Holeyland via Miami Canal.  Determine amount of Miami Canal
C   runoff backpumped if LOK is low and no water supply or environmental
C   needs in the WCA & LEC. IF backpumping is desired then redistribute
C   flows between S3 & S8.
C
            IF (runmode .eq. 'SIMUL') THEN
      CALL LAKE_NONREG_WCA(DELLEC,CRMIN,VOLWTR,NUMD ,WSNDE1,WSNDE2,
     $WSNDE3,WSNDES,WSNDESE,WSNEEDL8,DMDS4 ,dmds236,s65ernff,
     $dmdsemind,jday,pctmet ,ws_from_lo_s6,s236so,rlok_to_sa1,
     $outflow_from_sta2_for_ws ,runoff_298_districts,wsuply ,tcnsrnff,
     $divers,out_to_wca1_frm_sta1w ,out_to_wca1_frm_sta1e,runoff_south ,
     $ifacteaa,stgref_asr,stgref,ws_from_lo_s5a ,isem_flg,sem_dmnd_orig,
     $dmnd_sugh,ws_frm_lok_to_semcyp ,ws_frm_lok_to_sugh,dmnd_sugh_orig,
     $capac_semws,capac_sugh ,stgref_asr_rec,runoff_s7,runoff_s150,
     $avg_lokstage,stagelo_beg ,dmnd_nnr_aft_resasr,
     $res_outflow_to_basin ,avg_sim_wca_stg_floor,sem_dmnd_met_bef_lok,
     $ires_no ,water_supply_to_holyl,lok_criteria_wca1_lok_ws,
     $climate_index ,norstorrf, norstorpet,frac_env_met,
     $tcrnbsres_to_lok ,frac_thru_s8_sta34,ws_to_roten_frm_lok,
     $fplresws ,ws_to_sta34_frm_lok,rloc_excess_for_ws,
     $max_days_mean_loktasr ,sumrain,tot_flw_to_c9r_frm_nlkblt,
     $ro_factor,sprev_res,s1 ,dmnd_lec_s5a,dmnd_lec_s6,dmnd_lec_sa2,
     $stg_backflw_stl ,stgref_pulse_bot,stgref_nstor_to_lok,
     $stgref_lok_to_nstor ,vol_flow_res,runoff_south_bef_divers,
```

```
     $stgref_lok_to_eaasurgt ,stgref_regtoest,kfloref,nmdays_lok,
     $ws_from_acmres_lwdd,issm_cutb_opt_bcyp,pot_del_to_eaa_basin,
     $alocnnr_potdel,total_lec_dmnd_lok,issm_cutb_opt_brghton,
     $g94abcapac,rlvseep_frm_cnl,ws_bkp_cap,
     $IWSBPCNTR,MAXBPCNTR,iwsbkpwsta,bflo_cap_l8,
     $iroute_to_wca3a_frm_holy,igravity_s8_nosta,total_small_afact,
     $runoff_to_sta,s4bsn_to_lok,uisrnff,iday_lok_in_ssm,bmp_ssm_opt,
     $stgref_state_eaares,iopt_env_eaar_thru_g372)
c
C   LAKE_REG_WCA
C   Calculates the WS and regulatory discharges from LOK to the WCAs.
C   Management of excess water from LOK to WCAs and Caloosahatchee and
C   St Lucie Estuary based on regulatory releases from LOK. Computes
C   EAA Conveyance capacity for neutral case for Miami and NNR Canals
C   based on options input as a function of zones of LOK regulation
C   schedule.  Also calculates total limit on EAA RO and WS plus
C   regulatory flows (TFLOWS#) and (S6&S5A) discharge to WCA1/S7
C   discharge to WCA2A/S8 discharge to WCA3A plus determine CEVERG_NEEDS
C   when LIMIT_REG_REL_GLADES = 'SCHED'. Then routes appropriate portion
C   of RO into STAs.  Volume storage above reg from this routine this
C   will be a limit for regulatory releases through Caloosahatchee and
C   St Lucie Canals. Inject LOK water into proposed ASR system when LOK
C   stage is sufficiently high and retrieve from ASR when sufficiently
C   low.  Uses NORTHLOK_RES routine - computes volume of water to be
C   routed to north storage reservoir from LOK or taken from north
C   storage to LOK based on hydrologic condition. Then backpump
C   Caloosahatchee reservoir water into LOK if reservoir stage is above
C   threshold at prescribed rate. Compute regulatory releases to St
C   Lucie and Caloosahatchee Rivers (REGS308 & REGS77). Release steady
C   flow based on LOK stage.
c
              CALL LAKE_REG_WCA(volwtr,finstor,finstor_res,NUMD ,dmds4,
     $        dmds236,s65ernff,dmdsemind,jday,divers ,
     $        lo_reg_zone,lok_supp_day_c43est,bflo_cap_c43est,
     $        runoff_south ,lok_supp_day_c44est,bflo_cap_c44est,
     $        ifacteaa,stgref_asr ,stgref,stgref_asr_rec,hres01,hres02,
     $        qin_max01,qin_max02 ,ares01,ares02,supp_day_c43est_tmp,
     $        supp_day_c44est_tmp ,rlokreg2res1,rlokreg2res2,reg2est1,
     $        reg2est2,flow_asr_inj ,prev_avg_lokstg,avg_lokstage,
     $        max_days_mean_loktasr,rf_factor ,rmetvol_res,
     $        no_of_hurr_pred,climate_index,norstorrf, norstorpet ,
     $        tcrnbsres_to_lok,ws_to_roten_frm_lok,fplresws,
     $        res_seep_factor ,tot_cnl_dmd_asr,ro_factor,tcnsrnff,
     $        sprev_res,s1 ,dmnd_lec_s5a,dmnd_lec_s6,dmnd_lec_sa2,
     $        stg_backflw_stl ,stgref_pulse_bot,stgref_nstor_to_lok,
     $        stgref_lok_to_nstor ,stlro,calro,vol_flow_res,
     $        runoff_south_bef_divers,rstgpls ,stgref_lok_to_eaasurgt,
     $        stgref_regtoest,kfloref,dellec,ibeg_mth_lokdev,
     $        iend_mth_lokdev,ibeg_day_lokdev,iend_day_lokdev,
     $        iregcals79opt,rmthly_rf_lok,cfsoth1,
     $        cfsoth_litz1,cfsoth_res1,res_struc_inflw,day_rain_nsm,
     $        isum_nsm,minx_nsm,maxx_nsm,rf_factor_lok,ro_factor_lok,
     $        iopt_drawdown,ws_to_sta5e,s4bsn_to_lok,uisrnff,
     $        stgref_state_eaares,rlokreg_to_l8,capacc10a_bak)
c
c             print *,stage_res(1169,1),' AFT LAKE REG'
              prev_avg_lokstg = avg_lokstage
```

```
               total_flow_g404 = total_flow_g404 + ws_frm_lok_to_semcyp
c
c   Route excess water to res/ASR system
c
               DO ib = 1,neaabsn
                  DO is = 1,ninstr(ieaa_res_asr(ib))
                     DO k = 1,NINDPTS(ieaa_res_asr(ib),is)
                        pond(iupstrm(ieaa_res_asr(ib),is,k)) = pond
     $                  (iupstrm (ieaa_res_asr(ib),is,k)) +
     $                  flow_to_eaa_res_rnff(ib)*86400. /(ninstr
     $                  (ieaa_res_asr(ib))*NINDPTS(ieaa_res_asr(ib),is)
     $                  *GDAR)
                     END DO
                  END DO
                  DO kk = 1,no_of_res_reg_frm_lok(ib)
                     DO is = 1,ninstr(ieaa_res_asr_reg(ib,kk))
                        DO k = 1,NINDPTS(ieaa_res_asr_reg(ib,kk),is)
                           pond(iupstrm(ieaa_res_asr_reg(ib,kk),is,k)) =
     $                     pond(iupstrm (ieaa_res_asr_reg(ib,kk),is,k))
     $                     +flow_to_eaa_res_reg(ib,kk) *86400./(ninstr
     $                     (ieaa_res_asr_reg(ib,kk)) *NINDPTS
     $                     (ieaa_res_asr_reg(ib,kk),is)*GDAR)
                        END DO
                     END DO
                  END DO
               END DO
            ENDIF
c
c
c   Recover water from EAA ASR wells to meet remaining EAA demands
c
            IF (runmode .eq. 'SIMUL') THEN
               DO irs = 1,nres_ws
                  DO js = 1,noutlets(irs)
                     IF (idshg_opt(ires_no(irs),js).eq. 1.and.
     $               stagelo_beg .ge. stgref_asr_rec .and. iasr_opt_eaa
     $               (ieaa_bsn(irs,js)) .eq. 1) THEN
C
C   WS_FROM_RES
C   Computes water supply discharges from reservoir to appropriate
C   destinations [to meet EAA demand from local facilities]. Water
C   supply to meet EAA demands from EAA reservoirs.  Called if ASRs
C   associated with EAA exist and no runoff from ASR wells. Also if
C   discharge routing option = 1 [discharge to meet EAA basin
C   demands] and stagelo < stgref_asr_rec  or iasr_opt_eaa = 1 [RO
C   not injected into ASR wells].  Determine  volume of water available
C   for each outlet structure.  This is processed after calling ASR. It
C   is performed twice.  First to meet EAA demands with excess water
C   stored in res/ASR systems within EAA if LOK is sufficiently low and
C   then to calculate output from appropriate reservoir for water supply
C   purposes in LEC service areas. Processed after calling LAKE_REG_WCA
C   and routing excess water to res/ASR system.  It is used to recover
C   water from EAA ASR wells to meet remaining EAA demands.
C
                        call ws_from_res(irs,js,ires_no(irs),ieaa_bsn
     $                  (irs,js),crmin ,sumrain,jday,isem_flg,dmnd_sugh,
     $                  dmnd_sugh_orig,capac_semws ,capac_sugh,
```

```
     $                  stgref_asr_rec,res_outflow_to_basin,
     $                  stagelo_beg ,pctmet,sem_dmnd_orig,0,
     $                  outflow_from_sta2_for_ws,wsndes ,wsndese,
     $                  runoff_to_sta,volwtr,frac_thru_s8_sta34 ,wsnde1,
     $                  wsnde2,wsnde3,res_outflow_env,
     $                  rloc_excess_for_ws ,tot_flw_to_c9r_frm_nlkblt,
     $                  ws_from_acmres_lwdd,issm_cutb_opt_bcyp,
     $                  pot_del_to_eaa_basin,alocnnr_potdel,
     $                  total_lec_dmnd_lok,iroute_to_wca3a_frm_holy,
     $                  iopt_env_eaar_thru_g372)
                     ENDIF
                  END DO
               END DO
               flow_from_asr_meet_rem_eaa_dmnd = 0.0
               IF (stagelo_beg .ge. stgref_asr_rec) THEN
                  flow_from_asr_for_ws_eaa_1 = 0.0
                  DO ib = 1,neaabsn-1
                     IF (iasr_opt_eaa(ib) .eq. 1) THEN
                        vol_in = min(iagflo(iag(ib+6),1),0) * 1.9835
                        vol_in = abs(vol_in)
                        DO iar = 1,nasr_to_eaa(ib)
                           call asr (asrnum(3,ib,iar,2),2,vol_in ,
     $                     bal_vol,1,tot_cnl_dmd_asr)
                           vol_in = bal_vol
                        END DO
                        flow_from_asr_for_ws_eaa_1 =
     $                  flow_from_asr_for_ws_eaa_1 +abs(min(iagflo(iag
     $                  (ib+6),1),0)) * 1. 9835- bal_vol
                        IF (iagflo(iag(ib+6),1) .lt. 0) THEN
                           iagflo(iag(ib+6),1) = - nint( bal_vol/1.9835)
                        ENDIF
                     ENDIF
                  END DO
                  flow_from_asr_meet_rem_eaa_dmnd =
     $            flow_from_asr_for_ws_eaa_1
               ENDIF
            ENDIF

c   Water conservation area outflows
C
C   WCA_AVAIL_STOR_DMND_SETUP
C   Calculates demand and storage = TNEED; TINFLOW; ADD_NEED;
C   be met TOTAL_OUTF_FRM_CNL. Calculate the total LEC surface water
C   needs to by each of the WCAs. Information is needed for pro-rating
C   of available supply if supply is not sufficient to meet the needs.
C   Note: water supply needs is based on OPT_L8_PROP and LWDD. Routine
C   determines total structure inflow for each WCA based on adding
C   ADDFLW and KFLO TO TINFLOW.  Then determines demands and available
C   water for conceptual pipelines simulated.  Finally available
C   regulatory releases 'AVVOLRG' and available water supply 'AVVOLWS'
C   are calculated.
C
            CALL wca_avail_stor_dmnd_setup(TINFLOW,add_need
     $      ,total_outf_frm_cnl,reg_rel_frm_lok,outflow_from_3a
     $      ,TNEED_ORIG,wsneedl8,dqu,avail_vol_pipel
     $      ,tneed_pipel,nsvarea,month,avvolws,avvolrg,climate_index
     $      ,avvol_rlok_to_sa1,ws_from_lo_s5a,ws_from_lo_s6
```

```
     $      ,regcoffset,LWSLO_floor,chdep,frf,ds_convey_c_name
     $      ,WSNDE1,WSNDE2,WSNDE3,WSNDES,WSNDESE,ibot_zone_regd,jday)
c
C   WCAOUT
C   Water conservation area outflows determines the discharge at the
C   outflow structure of the WCAs. The number of known flows and the
C   names of each flow are identified. Define the HW and TW stages for
C   each structure options IHWOPT/ITWOPT. These options are incorporated
C   so the model knows if the HW & TW of a particular structure is
C   represented by a canal or node.  Discharges are stored in the KFLO
C   array. Determines structure flows as a function of gw level and/or
C   canal stage; apply water supply, flood control and environmental
C   rules. Downstream needs are met via conveyance canal or grid cell.
C   Also includes special code unique for appropriate structures.
c
            CALL WCAOUT(CRMIN,WSNDE1,WSNDE2,WSNDE3,
     $       WSNDES,WSNDESE,WSNEEDL8,
     $       rlok_to_sa1,wsdel_to_c51w,prevqu,
     $       wsdel_to_l8s,total_add_supply_to_lwdd,
     $       wsdel_to_l8,jday,rain,volwtr,tinflow,
     $       outflow_from_3a,istr_count,
     $       avg_sim_wca_stg_floor,outflow_from_sta2_for_ws,
     $       istr_count_knflw,s31dsg_to_bay,tempf_volwtr,
     $       tempf_weircff,no_of_hurr_pred,
     $       struc_name_meas,iopt_lim_des_cap_wca,
     $       isdnpflw_for_slope_wca,isuppflw_for_slope_wca,
     $       avvol_rlok_to_sa1,avvolws,avvolrg,
     $       total_outf_frm_cnl,reg_rel_frm_lok,
     $       tneed_pipel,avail_vol_pipel,TNEED_ORIG,
     $       total_small_afact,reg_cap_fact,fnm,
     $       out_flow_east0)
c           print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT WCAOUT'
c           CALL WCAOUT(CRMIN,WSNDE1,WSNDE2,WSNDE3,WSNDES,WSNDESE,
c    $      WSNEEDL8 ,ws_from_lo_s6,rlok_to_sa1,wsdel_to_c51w,prevqu,
c    $      wsdel_to_l8s,total_add_supply_to_lwdd ,wsdel_to_l8,jday,
c    $      rain,volwtr,tinflow ,outflow_from_3a,ws_from_lo_s5a ,
c    $      istr_count,avg_sim_wca_stg_floor ,outflow_from_sta2_for_ws,
c    $      istr_count_knflw,NSVAREA ,s31dsg_to_bay,tempf_volwtr,
c    $      tempf_weircff,no_of_hurr_pred ,ds_convey_c_name,
c    $      climate_index,struc_name_meas,regcoffset,
c    $      iopt_lim_des_cap_wca,isdnpflw_for_slope_wca,
c    $      isuppflw_for_slope_wca)
c
c


            DO i = 1,nwcas
               DO j = 1,n_intcnl(i)
                  addflw(i,j) = 0.0
               END DO
            END DO
C
c   Place structure discharges from/to appropriate locations.
c     discharges include boundary structure flows and
c      simulated flows from earlier routines
c
c           print *,'  BEF KNFLOWS'
```

```
C   KNFLOWS
C   This subroutine distributes known flows to the appropriate canal or
C   surface ponding location. Seasonal adjustments to regulation
C   schedules are also made in this routine. Place structure discharge
C   from/to appropriate locations. Discharges include boundary structure
C   flows and simulated flows from earlier routines. IOPT specifies how
C   the flows are distributed. Uses KNOWN_FLOW_ROUTE_SPEC.DAT file which
C   contains the known flow specification data. These data are used to
C   route measured (historical) discharges or simulated discharges
C   computed in STRUC_LOK_IN_WCA_FLOW, LAKE_REG_WCA and WCAOUT routines
C   to appropriate locations specified in the data. Only used for
C   calibration.  Each structure contains two or more records of data,
C   depending on the number of destination points.
C
            CALL KNFLOWS (NUMD(MONTH),WSNDE1,WSNDE2,WSNDE3,WSNDES,
     $      WSNDESE,lvseep_to_c6,istr_count_knflw,TSEEP_DIVERS,
     $      g155_flow_from_3a,g155_pos,struc_name_meas,
     $      icnl_no_reg_seas,iweir_no_reg_seas,nreg_seas,
     $      weir_reg_seas,isuppflw_for_slope_kn,isdnpflw_for_slope_kn,
     $      rlvseep_to_cnl,total_small_afact,ratio_flow_s6_sta2e,
     $      ratio_flow_s6_sta2m,ratio_flow_s6_sta2w)
c           print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT KNFLOWS'
C
c
c storm drawdown
c
           IF (ndays_prestorm_drawdown.gt.0) THEN
             IF (nstorm_day(ndays_sim_sd).eq.1) THEN
               DO id = 1,idrwnres
                 CRSTELEV(ires_no_drawdown(id),istruc_no_res_drwdwn(id)
     &               ,2)= CRSTELEV_drwdwn(id)
                 STGMX(ires_no_drawdown(id)) = stgmx_drwdwn(id)
               ENDDO
             ELSE
               DO id = 1,idrwnres
                 CRSTELEV(ires_no_drawdown(id),istruc_no_res_drwdwn(id)
     &            ,2) = CRSTELEV2(ires_no_drawdown(id)
     &            ,istruc_no_res_drwdwn(id))
                 STGMX(ires_no_drawdown(id)) = STGMX2(ires_no_drawdown
     &           (id))
               ENDDO
             ENDIF
           ENDIF
c
            DO node = 1,isum(maxy+1)
            DO ir = 1,noresincell(node)
            ia =ires_index_in_cell(node,ir)
             IF (sfactor(ia).LT. sfactmin .AND. ires_small_sim(ia)
     &       .EQ. 'YES') THEN
               DO is = 1,noutstr(ia)
                 IF (res_out_type(ia,is).ne.'WSPLY '.and.res_out_type
     &           (ia,is).ne.'SEEPG') THEN
                  IF (nodepw(ia,is,1) .eq. idnstrm(ia,is)) then
                      node_dn = nodepw(ia,is,1)
C
C   ETCOMP
```

```
C   Map daily rainfall values from DAY_RAIN array and update ponding
C   with today's rainfall. Then ET calculation for each cell.  Computes
C   ET components, infiltration, ponding, recharge, etc. ET is
C   calculated based on adjustment coefficients for land use types
C   7/8/9.  Then KC iscalculated based on ETK; adjust coefficients;
C   ET_CAL_ADJ_FACT. Then GWD [distance from land surface to water
C   table] is calculated and used to determine the type of ET and
C   ET_in_res calculations. Used to update ETM and RES_ET for each node
C   and then calculate PONDING,VOLWTR,RAIN,RES_RAIN,ETC.
C
                      CALL ETCOMP (1,ET,NODE_DN,LUTYP(NODE_DN),JDAY ,
     $                depth_of_deficit,stage_res ,ET_outside_res,
     $                ia,ir,sfactor(ia),ells_in_res,0)
                      deptgw =amax1(ells(nodepw(ia,is,1))-h(nodepw(ia
     &                ,is,1)),0.0)* s(nodepw(ia,is,1))
                      IF (pond(nodepw(ia,is,1))-et+rchg(node_dn) .gt.
     &                 deptgw) THEN
                        tail_p = pond(nodepw(ia,is,1)) - deptgw - et
     &                   +rchg(node_dn) + ells(nodepw(ia,is,1))
                        adj_for_ovlf = 0.0
                        if (tail_p .gt. ells(nodepw(ia,is,1))+0.10
     &                  .and. rain(nodepw(ia,is,1)) .gt. 2.0) then
                          adj_for_ovlf = amin1(rain(nodepw(ia,is,1))/12.
     &                    * 0.50,tail_p-ells(nodepw(ia,is,1))-0.10)
                        endif
                        tail_p = tail_p - adj_for_ovlf
                      ELSE
                        tail_p = (pond(nodepw(ia,is,1))-deptgw-et
     &                  + rchg(nodepw(ia,is,1)))/s(nodepw(ia,is,1))
     &                  + ells(nodepw(ia,is,1))
                      ENDIF
                  ELSE
                      IF (iwmads(ia,is).gt.ntotres) then
                        IF (inodopt(ia,is,3).eq.1) THEN
                         node_dn = nodepw(ia,is,1)
                         call etcomp(1,ET,NODE_DN,LUTYP(NODE_DN),JDAY ,
     &                   depth_of_deficit,stage_res ,ET_outside_res,
     &                   ia,ir,sfactor(ia),ells_in_res,0)
                         deptgw=amax1(ells(nodepw(ia,is,1))-h(nodepw(ia
     &                   ,is,1)),0.0)* s(nodepw(ia,is,1))
                         IF (pond(nodepw(ia,is,1))-et+rchg(node_dn).gt.
     &                    deptgw) THEN
                           tail_p = pond(nodepw(ia,is,1)) - deptgw - et
     &                     + rchg(node_dn) + ells(nodepw(ia,is,1))
                         ELSE
                           tail_p = (pond(nodepw(ia,is,1))-deptgw - et
     &                     +rchg(node_dn))/s(nodepw(ia,is,1))
     &                     + ells(nodepw(ia,is,1))
                         ENDIF
                        ELSE
                          IF (ircnl(ia,is,1).le.nch) THEN
                            IF (nodcr(ircnl(ia,is,1)) .gt. 1) then
                              diff_canlr_stg_frm_ds_end = float
     &                        (nodcr(ircnl(ia,is,1))-NODEPOS_CANLR(ia,is
     &                        ,1))/float(nodcr(ircnl(ia,is,1))-1)
     &                        *hdc(ircnl(ia,is,1),iseas)
                            ELSE
```

```
                              diff_canlr_stg_frm_ds_end = 0.0
                            ENDIF
                             tail_p=chdep(ircnl(ia,is,1))
     &                       + diff_canlr_stg_frm_ds_end
                          ELSE
                             tail_p=crstelev(ia,is,2)
                          END IF
                        ENDIF
                      ELSE
                        tail_p=ells(nodepw(ia,is,1))+pond_in_res
     &                  (nodepw(ia,is,1),ir)
                      ENDIF
                    ENDIF
c                   if (resname(ia).eq.'S332BR')
c    &              print *,pond(nodepw(ia,is,1)),ells(nodepw(ia,is,1))
c    &                ,deptgw,H(nodepw(ia,is,1)),'  MMMM'
                    IF (wrexp(ia,is,2).eq.1.0) THEN
                       qoutmx_p(ia,is)=weircff(ia,is,2)*(stgmx(ia)-tail_p)
                    ELSE
                       IF (type(ia,is).eq.'GRAVITY') THEN
                         wcrsth=amax1(crstelev(ia,is,2),tail_p)
                       ELSE
                         wcrsth=crstelev(ia,is,2)
                       END IF
                       IF (wcrsth.gt.crstelev(ia,is,2)) THEN
                         qfactor=0.90
                       ELSE
                         qfactor=1.0
                       END IF
                       IF (stgmx(ia).gt.crstelev(ia,is,2)) THEN
                          qoutmx_p(ia,is)=weircff(ia,is,2)*(stgmx(ia)-
     &                    crstelev(ia,is,2))**wrexp(ia,is,2)*qfactor
                       ELSE
                          qoutmx_p(ia,is)=0.0
                       END IF
                       qoutmx_p(ia,is) = amin1(qoutmx_p(ia,is)
     &                 ,avvolgr_mx(ia,is))
                       IF (inodopt(ia,is,3).ne.1.and.ircnl(ia,is,1)
     &                  .le.nch) THEN
                         IF (itrgcnl(ia,is,1) .le. nch) THEN
                            if (nodcr(itrgcnl(ia,is,1)) .gt. 1) then
                              diff_stg_cnl_trg = float((nodcr(itrgcnl
     &                        (ia,is,1))-nodepos_rcnl_trg(ia,is,1)))
     &                        /float(nodcr(itrgcnl(ia,is,1))-1)*
     &                        hdc(itrgcnl(ia,is,1),iseas)
                            else
                              diff_stg_cnl_trg = 0.0
                            endif
                            dn_cnl_stg_trg = chdep(itrgcnl(ia,is,1))
     &                      +diff_stg_cnl_trg
                         ELSE
                           if (icell_trg_loc(ia,is,1) .gt. 0) THEN
                              dn_cnl_stg_trg = h(icell_trg_loc(ia,is,1))
     &                        + pond(icell_trg_loc(ia,is,1))
                           else
                              dn_cnl_stg_trg = -901.0
                           endif
```

```
                        ENDIF
                        IF (dn_cnl_stg_trg.lt.dwnstgmx(ia,is,1)+
     &                  offset_reg_res(ia,is,1).and.dn_cnl_stg_trg
     &                  .ge.dwnstgmx(ia,is,1)) THEN
                          qoutmx_p(ia,is)=1.0/offset_reg_res(ia,is,1)*
     &                    (dwnstgmx(ia,is,1)+offset_reg_res(ia,is,1)
     &                    -dn_cnl_stg_trg)*qoutmx_p(ia,is)
                        ELSE IF (dn_cnl_stg_trg.ge.dwnstgmx(ia,is,1)+
     &                   offset_reg_res(ia,is,1)) THEN
                          qoutmx_p(ia,is)=0.0
                        END IF
                      ENDIF
                      IF (iwmads(ia,is).lt. ntotres) then
                        qoutmx_p(ia,is) = amin1(qoutmx_p(ia,is)
     &                  ,tvolwtrl_p(iwmads(ia,is)))
                        tvolwtrl_p(iwmads(ia,is)) = tvolwtrl_p(iwmads(ia
     &                  ,is)) - qoutmx_p(ia,is)
                      ENDIF
                    ENDIF
                    volwtr(ia) = volwtr(ia)+qoutmx_p(ia,is)-qoutmx(ia,is)
c                   if (resname(ia).eq.'C4IMP ') print *,volwtr(ia)
c    &                ,qoutmx_p(ia,is),qoutmx(ia,is),'   ADJUST  ',tail_p
c    &                ,stgmx(ia)
                   ENDIF
                   ENDDO
                  ENDIF
                 volwtr(ia) = amax1(volwtr(ia),0.0)
                 ENDDO
                ENDDO
c
c  channel routing using iterative mass balance procedure
c
c           print *,h(469),pond(469),rchg(469),
c    $      ' BEF CHNLF'
c            print *,hdc(imatch_c4,iseas),'  BEF CHNLF'
C   CHNLF
C   Prior to exe storm drawdown is calculated for CRSTELEV & STGMX for
C   each IDRAWNRES.  Then ETCOMP is called for each NORESINCELL and
C   NOUTSTR with VOLWTR(IA) calculated for each IRES_INDEX_IN_CELL.
C   Set up operations.For canals that will be drawn down during
C   appropriate periods. Theninclude dual operations for special cases.
C   Calculates channel routing using iterative mass balance procedure -
C   calculate canal downstream stageand outflow, seepage and overland
C   inflow given canal upstream inflows and downstream stage from the
C   end of the previous day. Then calculate injection into ASR wells if
C   any (then call ASR.F). Establish equilibrium canal stage by
C   calculating canal budget components. Calculate overlandflow for each
C   reach (OVLFL) and total seepage (CGVOL) and injection into ASR wells
C   and outflow (cout) at downstream structure and discharge for canals
C   with daily varying slope. Then call CANL_DEP_STRUC_PARAM_SETUP.F
C   = Flow calculations- pump downstream no more than can be accepted
C   without aggravating flood conditions, if applicable.
C   CANL_DEP_STRUC_CAPAC_SETUP.F= Setting flow parameters for canal
C   structures.  GEN_CANL_DEP_STRUC_FLW.F=Routine breaks out COUT from
C   water supply, flood control, and maximum available storage in
C   reservoir receiving discharge from structure, if applicable. This
C   routine uses general code when  specified in the canal_struc_
```

```
C   specs.dat file.  SPEC_CANL_DEP_STRUC_FLW.F=calculate flows, pumpage,
C   water supply, backflow, water supply release overflow and environ-
C   mental flows. This routine uses general code when specified in the
C   canl_struc_specs.dat file.  At the end; printout monthly components
C   of canal budget to FC=72. Routine calculates mean channel depth,
C   seepage, downstream outflow, and overland inflow given inflows and
C   mean depth at the end of the previous day.
C
            CALL CHNLF (IDYMAX,RAIN,AVOL,CRMIN ,RFLS155A,JDAY,VOLWTR,
     $      PREVQU,FINSTOR,tinflow ,outflow_from_3a,head_offset,
     $      total_flow_to_mcnl ,rlist_lvname,stagelo_beg,stgref,
     $      finstor_res ,RMFLWS12,total_flow_s12,s12_reg_total,
     $      s12_env_total ,RMFLWS333,temp_volwtr,tempf_volwtr,
     $      rloc_seep_for_ws ,rlocal_ws_to_e1,rlocal_ws_to_e2,
     $      rlocal_ws_to_e3 ,rlocal_ws_to_es,rlocal_ws_to_ese,
     $      ws_to_l8_via_s352 ,convey_canal_names,nconvey_total,pctmet ,
     $      temp_weircff,tempf_weircff,rloc_excess_for_ws ,
     $      rloc_excss_to_canl_next_step,dischg_to_bay_s29_frm_c11r ,
     $      s333_reg_lec,tot_cnl_dmd_asr,tot_flw_to_c9r_frm_nlkblt ,
     $      flow_to_cpbres_frm_e1,flow_to_c9r_frm_c9 ,
     $      flow_to_lbn_frm_c11,rnlkblt_to_s29 ,flow_to_ca1_frm_acmebsn,
     $      dellec,s333_reg_lec_abve_sch,s333_reg_lec_bel_sch,
     $      total_small_afact,ndsglim,nodedlim,fl_trig_constrt,
     $      struc_name_meas,node_wd,rmean_cnl_bot_elev,
     $      flow_resistvty_coef,iwuppflw_for_slope,iwdnpflw_for_slope,
     $      isuppflw_for_slope,isdnpflw_for_slope,rf_drawdwn_dph,
     $      c304_out_to_marsh,hdrop_fact,hdrop_max,rlvseep_to_cnl,
     $      s356_to_cell,s356_to_l29,is334_open_flg,
     $      frac_avail_cap_s333,frac_avail_cap_s334,weir_name,
     $      s333s334rg_offset,iroute_to_wca3a_frm_holy,stage_g3273,
     $      HWSEEP_F_L30,HDSEEP_F_L30,ilv_f_l30,no_of_l30spnod,
     $      icol_f_l30,irow_f_l30,rlvseep_to_l30,ipos_lv_l30,
     $      daily_seep,seepmon,tot_avl_l8_bpit_res_m1q,
     $      flow_to_c9r_frm_c11,hdrop_min,add_dmndagl8,
     $      n_gated_grav_outlets_fc,n_pump_outlets_fc,
     $      rmax_dschg_coef,isupp_at_ups_end,out_flow_east,
     $      out_flow_east0,iroute_acme_fc_to_sta1e_opt,
     $      i_canal_rf_chg_ops,frac_s319_sta1ew,frac_s319_sta1ee,
     $      iroute_l8rnf_to_sta_opt)

c            print *,h(469),pond(469),rchg(469),'  AFT CHNLF'
            DO node = 1,isum(maxy)
               daily_chg_dpth_ovl_vol(NODE) = 0.0
            END DO
C
C     CALCULATE OVERLAND FLOW
C
c           print *,'  BEF OVLNF'
C   OVLNF
C   Calculate overland flow for each cell.  Also calculate vertical flux
C   (infiltration and percolation). Impose boundary conditions for OVRLF
C   on surface water part of ENP. Assume gradient of water level to be
C   the same as land surface if slope of land surface is downward toward
C   west coast. Calculate flows in west-east direction.  The first
C   iteration is north-south direction. Second iteration when BSN(NODE)
C   .ne. BSN(NODE2) there is an intervening levee them don't calculate
C   OVRLF.  Calculate volume of OVRLF using Manning's equation then make
```

```
C   sure this amount does not cause gradient to reverse sign. Calculate
C   OVRLF correction for estimate of stage in reservoir whose outflow is
C   going into same cell in which reservoir is located and OVRLF flow is
C   allowed to/from cell. For non-EAA cells calculate infiltration and
C   ET but do this in agarea for EAA cells.  [Calculate soil
C   infiltration & ET assuming no unsaturated zone]. For LECSA calculate
C   soil infiltration assuming unsaturated zone exists. Routine calls
C   LEC_ET_COMP.F. Note: not all RINFLT percolates - comp recharge with
C   LEC_ET_COMP.F.
C
            CALL OVLNF(JDAY,RAIN,depth_of_deficit,sumrain_unadj,
     $       total_small_afact,total_afact,res_struc_inflw,
     $       ofml2,rmin_ofml2,allig_alley_res_fact)
c           print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT OVLNF'
c           print *,h(469),pond(469),rchg(469),'  AFT OVLNF'
c
C   EAACOR
C   Correct storage in EAA basins from flows in excess of structure
C   capacity or for cutbacks in delivery to EAA - make stage adjustment
C   for US Sugar Ranch if treated separate from other EAA basins.
C   Distributes excess runoff and irrigation requirements generated in
C   the EAA water table management subroutine - agarea for each EAA
C   node.  Account when runoff from EAA exceeds removal capacity. Note:
C   make adjustment for US Sugar if separate basin is commented out. If
C   land is flooded - value of KMAX represents that of open water
C   surface. All LU's are assumed to be open water when the ponding
C   depth is > DMAX (arbitrary).  As ponding increases this could mean
C   that the pan coefficient grows or decreases. Then correct for demand
C   not met by regional system in meeting agricultural demands in L-8
C   basin (other than sugar cane).
c
            CALL EAACORR(SOLMDPH,supply_excess,isem_flg,jday ,
     $      tot_runoff_excess,add_dmndagl8,no_nonscane_irr_cells_l8)
c           print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT EAACOR'
c           print *,h(nodebf),pond(nodebf),ells(nodebf),solmx(nodebf)
c    $      ,'  AFT EAACOR'
c           print *,agp(1)/86400.,kflo(ifls3),kflo(ifls8)
c    $       ,tot_runoff_excess(1),iagflo(IAG(IS8BSN),1),'  MAIN'
C-----------------
c
            CALL EXCADJFRESSTG(sumrain,rain,dpthgw_for_rain,total_afact)
C
C--------------------
c  calculate mean ponding depth for the grid cells in each reservoir
c  to determine regional gw flow to/from reservoir
c-----------------------
            DO ia = 1,ntotres
               DO IY = MAXY,1,-1
                  DO IX = MINX(IY),MAXX(IY)
                     node = IX - MINX(IY) +  1+ ISUM(IY)
                     pond_prev_gw(node) = pond(node)
                     DO k = 1,nnodes(ia)
                        IF (node .eq. node_reserv(ia,k)) THEN
                           IF (sfactor(ia) .GE. sfactmin .or.
     &                     ires_small_sim(ia) .eq. 'NO ') THEN
```

```
                              gcell_mean_ponding_depth(node) =pond(node)
                           ENDIF
                        ENDIF
                     ENDDO
                  ENDDO
               END DO
            END DO
c
C
C-----------------
c        print *,'  BEF GWF'
C   GWF
C   Prior to execution of GWF, calculate mean ponding depth for the grid
C   cells in each reservoir to determine regional gw flow to/from
C   reservoir.  Calculate groundwater flows-computes groundwater head at
C   each node. NOTE: solve the groundwater equation. Also calculate
C   vertical flux (infiltration and percolation).
C   EAA CELL = USE_LEC_ET*LEC_NODE_IN_MODEL(J) =0 .OR. ISBN(J) = 40 .OR.
C   FRAC_IRR_AREA_T(J) = 0 vs non-EAA cell -> determines algorithm for
C   GWHEAD.  The H(NODE) & GWHEAD(NODE) calculated based on GET_TIDE()
C   which fetches the current cell's tide for today. Then establish
C   boundary conditions for groundwater calculations; constant head at
C   coastline; no flow at interior boundary nodes. Include any proposed
C   (current) curtain walls. Then find current transmissivity values for
C   grid points. Calculate harmonic mean of transmissivities. Estimate
C   gw flow vectors.  Adjust water levels in cells with small reservoirs
C   for regional gw flow. Reset correct ponding where H is above land
C   surface root zone for non-EAA & irrigation EAA cells.
C
         CALL GWF(SOLMDPH,JDAY,rchg_gw)
c          print *,h(469),pond(469),rchg(469),'  AFT GWF  '
c         print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT GWF'
c         print *,h(nodebf),pond(nodebf),ells(nodebf),solmx(nodebf)
c    $      ,'  AFT GWF'
c
         DO ia = 1,ntotres
            DO IY = MAXY,1,-1
               DO IX = MINX(IY),MAXX(IY)
               node = IX - MINX(IY) +  1+ ISUM(IY)
                  DO k = 1,nnodes(ia)
                     IF (node .eq. node_reserv(ia,k)) THEN
                        IF (sfactor(ia) .GE. sfactmin.or.ires_small_sim
     $                  (ia) .eq. 'NO ') THEN
                           diff_pond = pond_prev_gw(node)-pond(node)
                           rinfilt_adj = amax1(diff_pond,0.0)
                           gcell_mean_ponding_depth_end(node)
     $                     =amax1(pond(node)-rchg_gw(node)+infilt(node)
     $                     -rinfilt_adj,0.0)
c
                           daily_seep_loss_res(node) =
     $                     gcell_mean_ponding_depth(node) -
     $                     gcell_mean_ponding_depth_end(node)
                           IF (H(node)+POND(node) .GT. ELLS(node)) THEN
                              seep_loss_res(node) =daily_seep_loss_res
     $                        (node) + seep_loss_res(node)
                              seep_loss_res_cell(node) =
```

```
     $                        daily_gw_loss_cell(node) +
     $                        seep_loss_res_cell(node)
                           ENDIF
                        ELSE
                           daily_seep_loss_res(node) =
     $                     daily_gw_loss_cell(node)
                        ENDIF
                     ENDIF
                  END DO
               END DO
            ENDDO
         END DO
c
c
      DO ia = 1,ntotres
         iresnum = ia
         IF (sfactor(iresnum) .LT. sfactmin .AND. ires_small_sim
     $   (iresnum) .EQ. 'YES' .AND. nnodes(iresnum) .gt. 1) THEN
C   SMALL_RESERV_OVLNF
C   Calculate overland flow through multi-cell narrow reservoirs. OVLNF
C   occurs if mean depth (HM) of flow > detention depth (DETEN_IN_RES).
C   Then calculate OVRLF(NODER,IDIRECT(IRESNUM,IN))
C   += SIGNR*AMIN1(VOLOV,OVMAX) based on upstream head (HU) > downstream
C   head (HD) else  OVRLF(NODER,IDIRECT(IRESNUM,IN))=
C   OVRLF(NODER,IDIRECT(IRESNUM,IN))- SIGNR*AMIN1(VOLOV,OVMAX)
C
            call small_reserv_ovlnf(iresnum,
     $      ells_in_res,pond_in_res,stage_res)
C
C   SMALL_RES_GW_FLOW
C   Solve groundwater equation/ calculate volume of groundwater across
C   cell boundary face/adjust up/down stream reserv stages. First fine
C   current transmissivity values for grid cells. Then calculate volume
C   of groundwater flow across cell boundary face. Finally, adjust
C   upstream and downstream reservoir stages [POND_IN_RES & STAGE_RES
C   based on HMVOL = volume of gw flow across cell boundary face].
C
            call small_res_gw_flow(iresnum,ells_in_res,stage_res,
     $      pond_in_res,prev_stage_res,aqperm_in_res,
     $      AQDEP_in_res,nca_r)
         ENDIF
      ENDDO
c
      IF (ntotres .GT. 0) THEN
c
c  calclulate structural outflows from reservoirs/STAS
c
c        nodebf1 = 24 - minx(13) + 1 + isum(13)
c        nodebf2 = 24 - minx(14) + 1 + isum(14)
c        print *,stage_res(nodebf1,1),stage_res(nodebf2,1),' BEF RESOUT'
C
C   RESOUT
C   Calculate structural outflows from reservoirs/STAs. Calculate flood
C   control releases from reservoirs and water supply releases from LOK
C   to reservoirs. XEQ only if NTOTRES > 0. Calculation adjusted in
C   estimate of reservoir stage due to difference in actual reservoir
C   area & total area of grid cells in which reservoir is located.
```

```
C   Calculations are based on if reservoir is large enough not to treat
C   as separate entry in grid domain - area adjustments are made. Then
C   calculate available storage in reservoir to be injected into ASR.
C   These calculations are based on separate entity status. Inject
C   excess water from reservoir to ASR [call ASR.F]. Adjust water levels
C   in reservoir based on separate entity status.  Recover water from
C   reservoir based on separate entity status. Additional calculations
C   based on INODOPT -  inflow/outflow recipient structures option.
C   Compute equilibrium stage for grid cell - reservoir system
C   [RESERV_CELL_EQUIL_STG.F] AVVOLGR is calculated based on separate
C   entity status. Determine if criteria are met for environmental water
C   supply releases from LOK to WCA3A & outflow from grid cell followed
C   by canal. Calculate hydraulic detention time for reservoirs. Route
C   total outflow from Rotenberger to appropriate area.
C
         CALL RESOUT(VOLWTR,SUMRAIN ,out_to_wca1_frm_sta1w,
     $   out_to_wca1_frm_sta1e ,outflow_to_canal_roten,outflow_to_marsh,
     $   ifacteaa ,convey_canal_names,rstgpls,stgref_asr,sem_dmnd_orig ,
     $   rloc_excss_to_canl_next_step,dischg_to_bay_s29_frm_c11r ,
     $   wcaenv,tot_cnl_dmd_asr,trig_wca_stg,regstg_wca ,rain,
     $   flow_to_cpbres_frm_e1 ,offset_to_sched,
     $   iopt_for_semcyp_prior_roten ,flow_to_c9r_frm_c9,
     $   flow_to_lbn_frm_c11 ,tot_flw_to_c9r_frm_nlkblt,rnlkblt_to_s29 ,
     $   flow_to_ca1_frm_acmebsn,ws_from_acmres_lwdd,total_afact,
     $   total_small_afact,nrgtrg,noder_trg,stgr_trg,tvolwtrl,
     $   avvolgr_mx,tvolwtrl_p,no_canl_in_cell,icanl_no_in_cell,
     $   ipos_cell_in_canl,rindiv_canl_area,total_cell_stor,
     $   frac_s319_sta1ew,frac_s319_sta1ee,ratio_flow_s6_sta2e,
     $   ratio_flow_s6_sta2m,ratio_flow_s6_sta2w)
c         print *,h(nodeag),solmx(nodeag),pond(nodeag),ells(nodeag)
c    $      ,'  AFT RESOUT'
c        print *,stage_res(nodebf1,1),stage_res(nodebf2,1),' AFT RESOUT'
c        print *,h(469),pond(469),rchg(469),'  AFT RESOUT'
C
         DO IA = 1,ntotres
            IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &      (ia) .eq. 'NO ') THEN
C
C   LARGER_RESERV_STOR
C   Calculate available storage for flow into reservoirs for next time
C   step and determine the maximum available storage for each reservoir.
C   It uses stage & ells for each model node not just for the reservoir
C   to initially calculate VOLWTR.
C
               CALL larger_reserv_stor(IA,VOLWTR,SUMRAIN,QOUTMX
     &         ,total_afact,deltastor,prev_stage_res,tvolwtrl
     &         ,iseas)
            ELSE
C
C   SMALL_RES_STOR
C   Determine maximum available storage for each small reservoir treated
C   as a separate entity in model grid domain. This routine does not
C   adjust for overland flow but it does in the outflow structures. This
C   is based on res_out_type which is for each reservoir and OUTFLOW
C   structure, the purpose of OUTFLOW thru each structure.  If true then
C   QOUTMX is  calculated and added to volwtr ELSE QOUTMX = 0.0.
C   Processed at the beginning of each times step & before SPECSETUPFDSS
```

```
C
               CALL small_res_stor(IA,VOLWTR,QOUTMX,sumrain,total_afact,
     &         prev_stage_res,deltastor,tvolwtrl,avvolgr_mx,iseas)
            ENDIF
         ENDDO
      ENDIF

            WRITE (67,'(3I5,8(1X,F9.2),2(1X,F6.3))') iyear+ifyr-1,
     $        month,iday,sem_cyp_tot_dmnd(jday),sem_cyp_dmnd_local
     $        (jday),sem_dmnd_orig,outflow_from_roten_to_semcyp*1.9835,
     $        outflow_from_sta6_to_semcyp*1.9835, ws_frm_lok_to_semcyp
     $        *1.9835,demand_semind,dmdsemind(jday),pump_def_semhlywd,
     $        pump_semhlywd
C
C   SPECSETUPFDSS
C   Accounting of special volumes on a daily basis to be put into DSS
C   output file. This subroutine calculates the total water supply and
C   regulatory discharges from LOK to the WCAs. Initially set up
C   separate components of flow for S5A2;S5A3;S5A4 & LOKREG2STLUC /
C   LOKREG2CALOOS. Then WS, DMND, RO, & WEIR FLOW. Then variable groups
C   are loaded given if conditions.  Minimum flows and levels variables
C   are written out to (106) - DAILY_MINLVL_SPECS.DAT. Written in DSS
C   form using STRFLW.
C
        CALL SPECSETUPFDSS(lok2stluc_est,lok2stluc_irrig
     $,lok2caloos_est,lok2caloos_irrig,holyinq,dmds4,dmds236
     $,s235inf,dmdsemind,lvseep_to_l33,lvseep_to_c6
     $,total_eaa_supply_deliv,water_supply_to_holyl
     $,rlokreg2res1,rlokreg2res2,reg2est1,reg2est2
     $,vol_rec_from_cpbasr,tcnsrnff,tcrnbsres_to_lok
     $,total_reuse,ws_pipeline_3a,ws_pipeline_3a_s150
     $,ws_pipeline_2a,ws_pipeline_ca1,ws_pipeline_ca1_hill
     $,ws_pipeline_ca1_wpb ,s31dsg_to_bay,s140fc,fplresws
     $,total_reuse_res,rloc_seep_for_ws,ws_to_l8_via_s352
     $,rloc_excess_for_ws,total_reuse_ws,s333_reg_lec
     $,asr_rec_ws,ws_from_lo_s6,res_outflow_to_basin
     $,ws_from_lo_s5a,RFLS155A,outflow_to_canal_roten
     $,outflow_to_marsh,S236SO,ws_to_sta5,ws_to_sta6
     $,ws_to_sta34_frm_lok,g155_pos,dischg_to_s8_frm_sta34_adj
     $,dischg_to_s7_frm_sta34_adj,total_flow_g404,wsdel_to_l8
     $,total_add_supply_to_lwdd,wsdel_to_l8s,wsdel_to_c51w
     $,s12_reg_total,s12_env_total,ws_frm_lok_to_semcyp
     $,ws_frm_lok_to_sugh,outflow_from_roten_thru_s8_pres
     $,sem_dmnd_met_bef_lok,flow_from_asr_for_ws_eaa,cfsoth_litz1
     $,flow_from_asr_meet_rem_eaa_dmnd,tot_lok_to_eaares,cfsoth_res1
     $,flow_asr_inj,flow_from_asr_ws,iconvey_canal_no,ires_no
     $,idshg_opt,noutlets,istrctr_ws,divers,ws_to_roten_frm_lok,nsvarea
     $,asr_rec_ws_cnty,g155_flow_from_3a,isem_flg,nres_ws,jday,cfsoth1
     $,sta34_out_to_wca3ane_adj,ws_from_acmres_lwdd,vol_rec_from_acmasr
     $,rlvseep_to_l30,s4bsn_to_lok, uisrnff, rlokreg_to_l8)
C
C
C    IF LP(21) ACCEPTED MAKE DUMP OF A LINE
C             FOR ENP MODEL
C
C     IF( LP(21) ) CALL LINEOUT
C
```

```
c
C   DAILY_OUTPUT
C   Daily output to various ASCII files; outputs components of ASR
C   budget FC=65 LOK stage output to FC=76; Estimated flow to GWT
C   [EST_FLOW_TO_GWT] TO FC=131; rainfall formula (FRF) TO FC=141;
C   compute mean ponding for cells with separate reservoir(s); write
C   out water budgets for small reservoirs to (iunit_no_res_file);
C   write out NSM flow targets S12; S333; S355 (if applicable) to
C   FC=151; write out daily slopes for appropriate canals to FC=190;
C   write out levee seepage to FC=102.
c
      CALL DAILY_OUTPUT(stagelo,s12_reg_total,s12_env_total ,rmflws333,
     $lok_split_option,sumrain,n_cells,lo_reg_zone ,stagelo_res,
     $no_of_small_res_budg_files,iunit_no_res_file,deltastor,iseas,
     $pond_end,stage_g3273,ltotal,daily_seep,i_canal_rf_chg_ops)
c      print *,hdc(imatch_c4,iseas),'  AFT DAILY OUTPUT'

c
C   RESERVOIR_INIT
C   Initialize appropriate variables for reservoirs.  NOTE: For each
C   node DEPTH_OF_DEFICIT is calculated and ETCOMP is called. Then for
C   each cell and NORESINCELL then PREV_STAGE_RES(NODE,IR) =
C   STAGE_RES(NODE,IR)
C
      CALL RESERVOIR_INIT(volwtr,sumrain,SUMRAIN_unadj,jday ,
     $gcell_mean_ponding_depth,gcell_mean_ponding_depth_end ,rain,
     $prev_stage_res)

c
c   Change for LEC ET routine
C
      IF(use_lec_et .gt. 0) THEN
C
C   LEC_ET_SUM_MONTHLIES
C   ET calculations are summed into monthly quantities.
C   NOTE: Some monthlies are converted back to inches; initialize sums
C   to first day of the month; some monthly sums are done outside of
C   routine (do not sum here); write out monthly sums on last day of
C   month - do this for each ET (UNSAT, UNSAT_IRR, UNSAT_NONIRR,
C   PONDING and SAT,SUPPLY-BY USE TYPE).
C
         call lec_et_sum_monthlies(iyr,IMTH, numd(month),getfdd,
     $   rmonthly_loss_bmp,total_small_afact)
      ENDIF
c
c   Output grid cell based results on daily basis
c
      IF ( (DAY1 .AND. (initialize_annually_opt .eq. 0 .or.
     $ ifyr+iyear-1 .eq. ifyr.and.month.eq.1)) .AND. .NOT.
     $ LP(25) )  THEN
C
C   PRINTLP
C   Model sends data to various files based on LP flags setup in
C   gen_model_param.dat file.  Prints out desired arrays on a daily
C   GWRITE surface ponding [POND_END(NODE)] TO FC=41.
C   basis.  GWRITE ET total [ETM] TO ET_TOTAL.BIN [FC=42].
C   GWRITE rainfall [RFM] TO RAINFALL.BIN [FC=44].
```

```
C   GWRITE WELLS(NODE) TO PUMPAGE.BIN [FC=45].
C   GWRITE infiltration monthly [INFILT_MONTH(NODE)]
C   & percolation monthly [PERC_MONTH(NODE)]  TO FC=46.
C   GWRITE GWF  TO FC=49 HMVOL_DAILY (NODE,1,2) [based on direction]
C   TO FC=49.  GWRITE surface flows [OVLFLO_DAILY(NODE,1/2) [direction]
C   TO FC=43. GWRITE monthly total nodal surface water flow to canal
C   [RMONTHLY_OVLFLO_TO_CANAL(NODE)] TO FC=39.
C   GWRITE monthly total nodal gw seepage flow to canal
C   RMONTHLY_SEEP_TO_CANAL(NODE)] TO FC=35.
C   GWRITE monthly canal evaporation [RMONTHLY_CANAL_EVAP] TO FC = 38.
C   WRITE LOK RF,ET, and end of month storage TO FC=80.
C   WRITE monthly total levee seepage [SEEPMON] TO FC = 32.
C   Finally print out BAGP for each EAA basin & RMEAN_ACSTG for each
C   monitoring point & STAGELO [& STAGELO_RES IF lok_split_option +1].
C
         CALL PRINTLP (imth,IDAY,NUMD(MONTH),FINSTOR,getfdd ,
     $   rmonthly_loss_bmp,rmetvol_res,finstor_res ,lok_split_option,
     $   stagelo_res,rmthly_rf_lok,n_cells,total_small_afact,pond_end,
     $   total_cell_stor,ltotal,seepmon,asr_t_cell_volm)
         GO TO 40
c   Hydro end
      ENDIF
C
      IF ( LP(25) ) CALL PRINTLP (imth,IDAY,NUMD(MONTH), FINSTOR,getfdd,
     $rmonthly_loss_bmp,rmetvol_res,finstor_res ,lok_split_option,
     $stagelo_res,rmthly_rf_lok,n_cells,total_small_afact,pond_end,
     $total_cell_stor,ltotal,seepmon,asr_t_cell_volm)
   40 IF ( LP(19) ) CALL DAILY_OVLNF_OUT(IDAY,RUNTITLE)
      IF( lp(22) .AND. lp(23)) THEN
         CALL DSTRING(MONTH, IDAY, IYEAR+IFYR-1, BUFFER)
         DO NODE = 1, ISUM(MAXY+1)
            VALUES(NODE) = H(NODE)+POND(NODE)-ELLS(NODE)
         END DO
c
         call val2val_w_lok(1,stagelo,stagelo_res,isum_w_lok,
     $   minx_w_lok ,maxx_w_lok)
         CALL SETGRID(RUNTITLE,maxylok,isum_w_lok(maxylok+1),DX,DY,
     $   minx_w_lok,maxx_w_lok,isum_w_lok)
         CALL GWRITE(47, BUFFER, values_w_lok, IERR)
c
         CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX, MAXX,
     $   ISUM)
      END IF
c
      IF( lp(22) .AND. lp(21)) THEN
          CALL DSTRING(MONTH, IDAY, IYEAR + IFYR - 1,datetag)
c         IF (.NOT. LP(23)) THEN
             CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX,
     $       MAXX, ISUM)
c         ENDIF
          DO NODE = 1, ISUM(MAXY+1)
            VALUES(NODE) = et_sat(NODE)*12. + et_ponding(NODE)*12.
     $      + et_unsat_actual(node)*12.*(1.0-carfact(node)
     $      - total_small_afact(node))
          END DO
          CALL GWRITE(127,datetag, values, ierr)
      ENDIF
```

```
      DAY1 = .FALSE.
c   Put call to sum_trig_heads back in main.F
c
c   Call sum_trig_heads to sum up the heads at trigger cells
C
C   SUM_TRIG_HEADS
C   Keep track of canal and/or grid cell stage at desired locations used
C   for LEC water restrictions. Sum up the heads at trigger cells. Sums
C   simulated heads at trigger cells for a user-specified number of days
C   during the month. Sums heads in trigger cells if we're in the right
C   time during the month.
C
      IF(use_trigger .ge. 1) THEN
         call sum_trig_heads(numd(month))
      ENDIF
C
   50 CONTINUE
C
C     WRITE GRID CELL BASED OUTPUT
C
C
C   PRINTLP
C   Model sends data to various files based on LP flags setup in
C   gen_model_param.dat file. Output grid cell based results on daily
C   basis - prints out desired arrays.
C   GWRITE surface ponding [POND_END(NODE)] TO FC=41.
C   GWRITE ET total [ETM] TO ET_TOTAL.BIN [FC=42].
C   GWRITE rainfall [RFM] TO RAINFALL.BIN [FC=44].
C   GWRITE WELLS(NODE) TO PUMPAGE.BIN [FC=45].
C   GWRITE infiltration monthly [INFILT_MONTH(NODE)]
C   and percolation monthly [PERC_MONTH(NODE)]  TO FC=46.
C   GWRITE GWF  TO FC=49 HMVOL_DAILY (NODE,1,2) [based on direction]
C   TO FC=49. GWRITE surface flows [OVLFLO_DAILY(NODE,1/2) [direction]
C   TO FC=43.  GWRITE monthly total nodal surface water flow to canal
C   [RMONTHLY_OVLFLO_TO_CANAL(NODE)] TO FC=39.
C   GWRITE monthly total nodal gw seepage flow to canal
C   [RMONTHLY_SEEP_TO_CANAL(NODE)] TO FC=35.
C   GWRITE monthly canal evaporation [RMONTHLY_CANAL_EVAP] TO FC = 38.
C   WRITE LOK RF,ET, and end of month storage TO FC=80.
C   WRITE monthly total levee seepage [SEEPMON] TO FC = 32.
C   Finally print out BAGP for each EAA basin & RMEAN_ACSTG for each
C   monitoring point & STAGELO [& STAGELO_RES IF lok_split_option +1].
C
      IF (.not.lp(25)) CALL PRINTLP (imth,IDAY - 1,numd(month), FINSTOR,
     $getfdd,rmonthly_loss_bmp ,rmetvol_res,finstor_res,
     $lok_split_option ,stagelo_res, rmthly_rf_lok,n_cells,
     $total_small_afact,pond_end,total_cell_stor,ltotal,seepmon,
     $asr_t_cell_volm)

C
      IF((IMTH.EQ.IEM).OR.(MONTH.EQ.12))THEN
C       CALL YMDDAT(IYR,MONTH,NUMD(MONTH),104,EDATE,NDATE,IERR)
C   -1 = MEANS ENTIRE SIMQ2DSS WILL BE EXECUTED
C
C   Writes daily simulated structure discharges into output DSS file
C   'STR2X2.DSS'-- OVERWRITES. NOTE: STRFLW{structure flow points} is
C   initialized for each day and for each structure. Also processed at
```

```
C   end of monthly time step after sum_trig_heads.F
C
         CALL SIMQ2DSS(IFLTABS,SDATE,-1)
C
C   SIMCSTGDSS
C   Writes daily flows for simulated canal stages into output DSS file
C   'CANAL.DSS'. If pathnames already exist then it writes over them.
C   If not; then it appends these simulation flows at the end of the
C   DSS file. NOTE: CNLSTAGE is initialized to 0.0 for each day and
c   canal structure.
C
         CALL SIMCSTG2DSS(IFLTABC,SDATE,-1)
         DO it = 1,2
            DO node = 1,max_ncells
               IF (vol_ovlflo(node,iyear,it) .gt. 0.0) THEN
                  ratio_excess_ovlflo(node,iyear,it) = excess_vol_ovlflo
     $            (node ,iyear,it) / vol_ovlflo(node,iyear,it)
               ELSE
                  ratio_excess_ovlflo(node,iyear,it) = 0.0
               ENDIF
            END DO
         END DO
      ENDIF
c
c   WRMI, 12/04/92,  Call to SET_CUTBACK to
c
      IF(use_trigger .ge. 1) THEN
C
C
C   SET_CUTBACK
C   Evaluate recorded heads at trigger locations, identify severity of
C   water restriction and set appropriate cutback levels for the next
C   month.  If zone was in a cutback phase the previous month and if in
C   the dry season (Nov-May) then current zone must stay at least phase
C   1 this month.  Otherwise, initial cutback to 0 for zone - no
C   restrictions. NOTE: If head is less than trigger value for a given
C   phase we have violation.  Kick out of loop once it finds a
C   violation -- only LOK SSM (or SSM + dry season) will trigger
C   cutback next month (100) and only the dry season will.
C
         call set_cutback(iyr,month,numd(month),cnm,lok_ssm_cnt
     $      ,min_lok_ssm_cnt)
      ENDIF
c
c   WRMI <<
C
   60 CONTINUE
C
      CALL ZCLOSE(IFLTABS)
      call zclose(ifltabr)
      call zclose(ifltab1)
C
C   FINAL_OUTPUT
C   Final output at end of simulation.  Output mean # of days per year
C   volume limit for overland flow is reached. If LP(2) =print stage(H)
C   ponding(P for succeeding simulations (creates restart_output)
C
```

```
C
      CALL FINAL_OUTPUT(stagelo)
C
      timediff=dtime(timearray)
      WRITE(*,*)' CPU time: ',timediff/60.,' minutes'
C
      WRITE(*,*) ''
      WRITE(*,*) 'SFWMM Version %G%'
      WRITE(*,*) 'Terminated Correctly'
      END
```

map_lec_array_to_model.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: map_lec_array_to_model.F,v 1.4 2003/07/03 20:07:36 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/map_lec_array_to_model.F,v $
C   $Name: rel-5-5 $
C
C
C   Subroutine MAP_LEC_ARRAY_TO_MODEL WRITEs an LEC array to a model array,
C   and converts the units from inches to feet.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE map_lec_array_to_model (lec_array, model_array)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'

      INTEGER i, lec_node, model_node

      REAL lec_array(max_n_cells_lec), model_array(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      DO i = 1, max_ncells
         model_array(i) = 0
      END DO
      DO lec_node = 1, num_lec_nodes
         lec_array(lec_node) = lec_array(lec_node) / 12.
         model_node = model_node_in_lec(lec_node)
         model_array(model_node) = lec_array(lec_node)
      END DO
      RETURN
      END
```

map_lec_node_to_model.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: map_lec_node_to_model.F,v 1.4 2003/07/03 20:07:36 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/map_lec_node_to_model.F,v $
C   $Name: rel-5-5 $
C
C
C  Subroutine MAP_LEC_NODE_TO_MODEL checks to see IF a model node is in the
C  LEC developed area.  If so, the model node number is mapped to the LEC
C  node.  Array model_node_in_lec contains the model node number associated
C  with each LEC node. Array lec_node_in_model contains the model node
C  number for the LEC node.
C  cjn>> NOTE CORRECTION TO DESCRIPTION!
C  with each LEC node. Array lec_node_in_model contains the LEC node
C  number for each model node.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE map_lec_node_to_model

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'

      INTEGER lec_node, model_node, x, y

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   model_node is the node number of the model node.
C   lec_node is the node number of the LEC node

      DO i = 1,max_ncells
         lec_node_in_model(i) = 0
```

```
 END DO
 DO i =1,550
    model_node_in_lec(i) = 0
 END DO
 DO y = 1, maxy
    DO x = minx(y), maxx(y)
       IF ((x .GE. min_x_lec(y)) .AND. x .LE. max_x_lec(y)) THEN
       model_node = isum(y) - minx(y) + x + 1
       lec_node = isum_lec(y) - min_x_lec(y) + x + 1
       lec_node_in_model(model_node) = lec_node
       model_node_in_lec(lec_node) = model_node
    ENDIF
 END DO
 END DO
 num_lec_nodes = isum_lec(maxy) + max_x_lec(maxy) - min_x_lec(maxy)
$ + 1
 RETURN
 END
```

mgateop.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: mgateop.inc,v 1.4 2003/07/03 20:08:02 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/mgateop.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
c COMMON block for structure max. gate opening data
      common /mgateop/ dsstg(22,30),usstg(20,22,30),strgo(20,22,30),
     &ntails(30),n_strgos(30)
```

minfllvl.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: minfllvl.inc,v 1.4 2003/07/03 20:08:02 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/minfllvl.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /minfllvl/ LWSLO_floor_temp(max_n_wcas,max_n_minfllvl_loc),
     &logic_floor(max_n_wcas),
     &avg_wca_subarea_crit_stage(max_n_wcas,max_n_minfllvl_loc),
     &avg_wca_subarea_stg_disc_lok(max_n_wcas,max_n_minfllvl_loc),
     &index_lok_deliv(max_n_wcas,max_n_minfllvl_loc),
     &avg_wca_subarea_stg_strt_count(max_n_wcas,max_n_minfllvl_loc),
     &no_days_deliv_from_lok(max_n_wcas,max_n_minfllvl_loc),
     &index_ws_frm_outside_source(max_n_wcas,max_n_minfllvl_loc),
     &index_ws_from_lok(max_n_wcas,max_n_minfllvl_loc),
     &index_ws_frm_outside_src_canal(max_n_wcas,max_n_minfllvl_loc)
C------------------------
      logical LWSLO_floor_temp
```

mthly_unsat_et_irr_comp.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: mthly_unsat_et_irr_comp.F,v 1.11 2005/06/10 13:29:13 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/mthly_unsat_et_irr_comp.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine computes monthly sums for the net
C   irrigation demands and for the unsaturated zone ET
C   from irrigated areas in developed areas.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE mthly_unsat_et_irr_comp (numd,day_net_irr,
     & day_et_unsat_irr, day_et_poten, day_et_unsat_nonirr,
     & day_et_unsat, month, wfld, pump_def_semhlywd, pump_semhlywd)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'stat.inc'

      DIMENSION day_et_poten(max_ncells,31),
     & day_et_unsat(max_ncells,31), day_et_unsat_irr(max_ncells,31),
     & day_et_unsat_nonirr(max_ncells,31),
     & day_net_irr(max_ncells,max_n_type,31), numd(12)

      CHARACTER wfld(max_n_wells)*9

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C     Read one month of daily etu components and net irrigation
C     demands using the lec_et_time_series_read routine.
C     Compute monthly sums for ONLY the net irrigation demands and
C     for the unsaturated zone ET from irrigated areas.
```

```
C     (the monthly net irrigation demands & et_unsat_irr are needed
C     for computing reductions in ET via the est_etiu function).
C     The monthly sums for the actual ET and supply components, etc,
C     will be computed and written to binary output via the subroutine
C     lec_et_sum_monthlies.

      DO iday = 1,numd(month)

C     Read all 3 et components (pet,etu,etiu), compute etnu; and
C     read all 6 net irrigation components(landscape,nursery,golf,
C     ag_lowvol,ag_overhd,ag_other).

         CALL lec_et_time_series_read(iday)

C     save daily values in separate arrays

         DO lec_node = 1,num_lec_nodes
            model_node = model_node_in_lec(lec_node)
            L = model_node
            IF ( ibsn(L) .NE. 40 .AND. frac_irr_area_t(L) .GT. 0. ) THEN
               day_et_poten(L,iday) = et_poten(L)
               day_et_unsat(L,iday) = et_unsat(L)
               day_et_unsat_irr(L,iday) = et_unsat_irr(L)
               day_et_unsat_nonirr(L,iday) = et_unsat_nonirr(L)
               DO itype = 1,num_use_type
                  IF (irrig_area(L,itype) .GT. 0.) THEN
                     day_net_irr(L,itype,iday) = net_irr(L,itype)
                  ENDIF
               END DO
c     sum monthly totals for net irrigation demands and et_unsat_irr
               et_unsat_irr_month(L) = et_unsat_irr_month(L) +
     $         day_et_unsat_irr(L,iday)*(1.0-carfact(L))
               DO itype = 1,num_use_type
                  IF (irrig_area(L,itype) .GT. 0.) THEN
                     net_irr_month(L,itype) = net_irr_month(L,itype) +
     $               day_net_irr(L,itype,iday)*(1.0-carfact(L))
                  ENDIF
               END DO
            ENDIF
C END of test for ibsn().NE.40 and frac_irr_area_t().GT.0.
         END DO   !END loop for lec nodes
      END DO      !END day loop for et & nirr demand reads
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c set supply_month() equal to net_irr_month() IF the trigger module
c is off or IF no cutbacks are imposed for this month,
c ELSE, it will be set after reduce_wellq is CALLed
c
      IF (use_trigger .EQ. 0 .OR. .NOT. (lec_restrictions) ) THEN
         DO lec_node = 1,num_lec_nodes
            model_node = model_node_in_lec(lec_node)
            L = model_node
            DO itype = 1,num_use_type
               supply_month(L,itype) = net_irr_month(L,itype)
            END DO
         END DO
         pump_def_semhlywd = 0.0
```

```
         pump_semhlywd = 0.0
         DO iwell = 1,nwell
            supply_month_pws(locw(iwell)) = supply_month_pws(locw(iwell)
     $      ) + pump(iwell,month)
            IF (WFLD(iwell) .eq. '06-00001-') THEN
               pump_def_semhlywd = pump_def_semhlywd + pump(iwell,month)
               pump_semhlywd = pump_semhlywd + pump(iwell,month)
            ENDIF
         END DO
      ENDIF
      RETURN
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

mthly_variables_init.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: mthly_variables_init.F,v 1.16 2005/06/10 13:29:14 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/mthly_variables_init.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE INITIALIZES APPROPRIATE VARIABLES AT BEGINNING
C   OF EACH MONTH OF SIMULATION.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE mthly_variables_init (rmonthly_loss_bmp, rmetvol,
     & rmetvol_res, lok_ssm_cnt, crmin, mon_pet0,
     & rmthly_rf_lok, crmin0, crmin_bel_flr,innrc,
     & ofml2,rmin_ofml2,owpond2,owpond,asr_t_cell_volm)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'routc.inc'

      DIMENSION crmin(max_ncnls,max_n_seas), mon_pet0(max_ncells)
     &,crmin0(max_ncnls,max_n_seas),crmin_bel_flr(max_ncnls,max_n_seas)
     &,rmnstgs_c(max_ncnls,max_nostr_cnl,max_n_seas)
     &,rmxstg_c(max_ncnls,max_nostr_cnl,max_n_seas),ofml2(max_ncells)
     &,rmin_ofml2(max_ncells),owpond2(max_ncells)
```

```
     &,owpond(max_n_lu_type),asr_t_cell_volm(max_ncells)

      REAL mon_pet0

      DOUBLE PRECISION crmin,crmin0,crmin_bel_flr
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      QRNOFFM = 0.0
      QHLWSM = 0.0
      QWTSPM = 0.0
      IF( LP(20) ) THEN
         DO 10 I = MAXY,1,-1
            DO 10 NODE = ISUM(I)+1, ISUM(I+1)
               DO 10 J =  1, 2
                  OVLFLO (NODE,J) = 0.0
                  HMVOL(NODE,J) = 0.0
   10    CONTINUE
      ENDIF
c
      DO  node = 1,max_ncells
         rmonthly_ovlflo_to_canal(node) = 0.0
         asr_t_cell_volm(node) = 0.0
      END DO
c
      rmonthly_loss_bmp = 0.0
      RMETVOL = 0.0
      rmthly_rf_lok = 0.0
      rmetvol_res = 0.0
      lok_ssm_cnt = 0
c
      DRYSEAS = (MONTH .GE. 11) .OR. (MONTH .LE. 5)
      HURRPRED = (MONTH .GE. 8) .AND. (MONTH .LE. 11)
      IF (DRYSEAS) THEN
         iseas = 1
      ELSE
         iseas = 2
      ENDIF
c
C      ET  and  RAINFALL and WELL COUNTERS TO ZERO
C
      DO 20 I = MAXY,1,-1
         DO 20 L = ISUM(I) + 1,ISUM(I + 1)
c                  SINFLTM(L) = 0.0
            IF (use_lec_et .NE. 0) THEN
c
c     need to keep these statements to initialize monthly sum arrays
               et_unsat_irr_month(L) = 0.0
               etiu_reduction_t(L) = 0.0
               DO itype = 1, num_use_type
                  net_irr_month(L,itype) = 0.0
                  supply_month(L,itype) = 0.0
                  shortage_month(L,itype) = 0.0
                  etiu_unrestr(L,itype) = 0.0
                  etiu_restr(L,itype) = 0.0
                  etiu_reduction(L,itype) = 0.0
               END DO
c
```

```
            ENDIF
            ETM(L) = 0.0
            RFM(L) = 0.0
            WELLS(L) = 0.0
   20 CONTINUE
c
c  mimmick operations of s-39 to maintain lower water levels
c  in northern Broward county in 1989.AND.1990 - only for
c  Calibration run
c
      IF (runmode .EQ. 'CALIB') THEN
         IF ((ifyr+iyear-1 .GE. 1989 .AND. ifyr+iyear-1 .le. 1990)
     $      .OR. ifyr+iyear-1 .EQ. 1988 .AND. MONTH .GE. 10) THEN
            DO is = 1,2
               crmin(lhls,is) = 10.75
            END DO
         ELSE IF (ifyr+iyear-1 .GE. 1991 .AND. ifyr+iyear-1 .le. 1995)
     $   THEN
            DO is = 1,2
               crmin(lhls,is) = 12.50
            END DO
         ELSE IF (ifyr+iyear-1 .GE. 1996 .AND. ifyr+iyear-1 .le. 2000)
     $   THEN
            DO is = 1,2
               crmin(lhls,is) = 12.0
            ENDDO
         ELSE
            DO is = 1,2
               crmin(lhls,is) = 13.5
            END DO
         ENDIF
         IF (ifyr+iyear-1 .EQ. 1991 .OR. (ifyr+iyear-1 .EQ. 1992
     $       .AND. MONTH .LT. 9)) THEN
            DO jseas = 1,2
              DO  is = 1,nstrctr(l31s)
                rmnstgs_c(l31s,is,jseas) = 4.0
                rmxstg_c(l31s,is,jseas) = 4.5
              ENDDO
            ENDDO
         ELSE
            DO jseas = 1,2
              DO is = 1,nstrctr(l31s)
                rmnstgs_c(l31s,is,jseas) = rmnstg0(l31s,is,jseas)
                rmxstg_c(l31s,is,jseas) = rmxstg0(l31s,is,jseas)
              ENDDO
            ENDDO
         ENDIF
         DO jseas = 1,2
           DO is = 1,nstrctr(l31s)
              rmnstgs(l31s,is,jseas) = rmnstgs_c(l31s,is,jseas)
              rmxstg(l31s,is,jseas) = rmxstg_c(l31s,is,jseas)
           ENDDO
           DO is = 1,nstrctr(l31s)
              RMXSTG2(l31s,is,jseas) = rmxstg(l31s,is,jseas)
              RMNSTG2(l31s,is,jseas) = rmnstgs(l31s,is,jseas)
           ENDDO
         ENDDO
```

```
      DO is = 1,2
        crmin0(lhls,is) = crmin(lhls,is)
        CRMIN_bel_flr(lhls,is) = CRMIN(lhls,is)
      ENDDO
      call match('NNRFG',1,5,cnm,nch,5,innrfg)
      IF (ifyr+iyear-1 .EQ. 1991.or.(ifyr+iyear-1
$      .EQ. 1992 .and. month .le. 5)) THEN
         DO jseas = 1,2
           DO is = 1,nstrctr(innrfg)
             rmnstgs_c(innrfg,is,jseas) = 3.8
             rmxstg_c(innrfg,is,jseas) = 4.2
           ENDDO
         ENDDO
      ELSE
         DO jseas = 1,2
           DO is = 1,nstrctr(innrfg)
             rmnstgs_c(innrfg,is,jseas) = rmnstg0(innrfg,is,jseas)
             rmxstg_c(innrfg,is,jseas) = rmxstg0(innrfg,is,jseas)
           ENDDO
         ENDDO
      ENDIF
      DO jseas = 1,2
         DO is = 1,nstrctr(innrfg)
           rmnstgs(innrfg,is,jseas) = rmnstgs_c(innrfg,is,jseas)
           rmxstg(innrfg,is,jseas) = rmxstg_c(innrfg,is,jseas)
         ENDDO
         DO is = 1,nstrctr(innrfg)
           RMXSTG2(innrfg,is,jseas) = rmxstg(innrfg,is,jseas)
           RMNSTG2(innrfg,is,jseas) = rmnstgs(innrfg,is,jseas)
         ENDDO
      ENDDO
      IF ((ifyr+iyear-1 .EQ. 1992 .and. month .ge. 6) .or.
$      (ifyr+iyear-1 .EQ. 1994 .and. month .le. 9) .or.
$      ifyr+iyear-1 .EQ. 1993) THEN
         DO jseas = 1,2
           DO is = 1,nstrctr(innrc)
             rmnstgs_c(innrc,is,jseas) = 4.2
             rmxstg_c(innrc,is,jseas) = 4.5
           ENDDO
         ENDDO
      ELSE
         DO jseas = 1,2
           DO is = 1,nstrctr(innrc)
             rmnstgs_c(innrc,is,jseas) = rmnstg0(innrc,is,jseas)
             rmxstg_c(innrc,is,jseas) = rmxstg0(innrc,is,jseas)
           ENDDO
         ENDDO
      ENDIF
      DO jseas = 1,2
         DO is = 1,nstrctr(innrc)
           rmnstgs(innrc,is,jseas) = rmnstgs_c(innrc,is,jseas)
           rmxstg(innrc,is,jseas) = rmxstg_c(innrc,is,jseas)
         ENDDO
         DO is = 1,nstrctr(innrc)
           RMXSTG2(innrc,is,jseas) = rmxstg(innrc,is,jseas)
           RMNSTG2(innrc,is,jseas) = rmnstgs(innrc,is,jseas)
         ENDDO
```

```
         ENDDO
         IF ((ifyr+iyear-1 .EQ. 1982 .and. month .ge. 4) .or.
     $    ifyr+iyear-1 .EQ. 1983 .or. (ifyr+iyear-1 .EQ. 1984
     $    .and. month .le. 3)) then
            DO jseas = 1,2
              DO is = 1,nstrctr(ic51)
                rmnstgs_c(ic51,is,jseas) = 7.3
                rmxstg_c(ic51,is,jseas) = 8.0
              ENDDO
              crmin(ic51,jseas) = 7.0
            ENDDO
         ELSE
            DO jseas = 1,2
              DO is = 1,nstrctr(ic51)
                rmnstgs_c(ic51,is,jseas) = rmnstg0(ic51,is,jseas)
                rmxstg_c(ic51,is,jseas) = rmxstg0(ic51,is,jseas)
              ENDDO
              crmin(ic51,jseas) = 7.8
            ENDDO
         ENDIF
         DO jseas = 1,2
            DO is = 1,nstrctr(ic51)
              rmnstgs(ic51,is,jseas) = rmnstgs_c(ic51,is,jseas)
              rmxstg(ic51,is,jseas) = rmxstg_c(ic51,is,jseas)
            ENDDO
            DO is = 1,nstrctr(ic51)
              RMXSTG2(ic51,is,jseas) = rmxstg(ic51,is,jseas)
              RMNSTG2(ic51,is,jseas) = rmnstgs(ic51,is,jseas)
            ENDDO
         ENDDO
         DO is = 1,2
           crmin0(ic51,is) = crmin(ic51,is)
           CRMIN_bel_flr(ic51,is) = CRMIN(ic51,is)
         ENDDO
       ENDIF
c
c  account for hurricane Andrew in temporary change in landscape
c
         IF ((ifyr+iyear-1 .EQ. 1992 .and. month .ge. 8)
     $   .or. ifyr+iyear-1 .EQ. 1993 .or. (ifyr+iyear-1
     $   .eq. 1994 .and. month .le. 5)) then
            DO  I = MAXY,1,-1
              DO  NODE = ISUM(I)+1, ISUM(I+1)
                IF (I .GE. 12 .AND. I .LE. 30) THEN
                   IF (LUTYP(NODE) .EQ. 17 .OR. LUTYP(NODE) .EQ. 18
     $             .OR. LUTYP(NODE) .EQ. 21 .OR. LUTYP(NODE) .EQ. 23)
     $             THEN
                     OFML2(NODE) = OFML(LUTYP(NODE),1)*0.40
                     rmin_ofml2(NODE) = 0.2
                     owpond2(NODE) = 2.5
                   ENDIF
                ENDIF
              ENDDO
            ENDDO
         ELSE
            DO  I = MAXY,1,-1
              DO  NODE = ISUM(I)+1, ISUM(I+1)
```

```
                IF (I .GE. 12 .AND. I .LE. 30) THEN
                   IF (LUTYP(NODE) .EQ. 17 .OR. LUTYP(NODE) .EQ. 18
     $             .OR. LUTYP(NODE) .EQ. 21 .OR. LUTYP(NODE) .EQ. 23)
     $             THEN
                     OFML2(NODE) = OFML(LUTYP(NODE),1)
                     rmin_ofml2(NODE) = rmin_ofml(LUTYP(NODE))
                     owpond2(NODE) = owpond(LUTYP(NODE))
                   ENDIF
                ENDIF
              ENDDO
            ENDDO
         ENDIF
c
c      initialize pws monthly arrays
      DO i = 1,max_ncells
         demand_month_pws(i) = 0.0
         supply_month_pws(i) = 0.0
         shortage_month_pws(i) = 0.0
      END DO
c
c
c note: demand_month_pws() are the unadjusted pumpage
c       totals (over all wells in each node);
c       pumpage for industrial and self-supplied wells
c       are assumed not to be subject to cutbacks
      DO iw = 1, nwell
         demand_month_pws(LOCW(IW)) = pump(iw,month) + demand_month_pws
     $   (locw(iw))
      END DO
c
      DO iz = 1, netzon
         mon_pet0(iz) = 0.0
      END DO
      RETURN
      END
```

munic_well_pump_setup.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: munic_well_pump_setup.F,v 1.9 2005/06/10 13:29:14 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/munic_well_pump_setup.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE READS MUNICIPAL WELLFIELD PUMPAGES WHICH INCLUDE
C   RESIDENTIAL AND INDUSTRIAL AREAS AND SETS UP PARAMETERS FOR
C   DETERMINATION OF THE CONTRIBUTION OF PUMPAGES TO RECHARGE (RCHG)
C   TERM OF GROUNDWEATER EQUATION AT APPROPRIATE GRID CELLS.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE munic_well_pump_setup (welldat_opt, wfld)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'

      CHARACTER char_frac_pump*6, welldat_opt*5, wfld(max_n_wells)* 9,
     & wfld_ind_rss(max_n_wells_rss)*9

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   READ THIS YEARS PUMPAGE FIGURES

      IF( welldat_opt .EQ. 'TIMEV') THEN

         READ(18,'(I4,1x,F6.2)') NWELL,frac_pump
         BACKSPACE 18
         READ (18,'(5x,A6)') char_frac_pump
         IF (char_frac_pump .EQ. '      ') THEN
            frac_pump = 1.0
            write(*,10) ifyr + iyear - 1
```

```
         ENDIF
         DO 20 IW=1,NWELL
            READ(18,60) WFLD(IW),IXP,IYP,(PUMP(IW,MONTHP),MONTHP=1,12)
            DO MONTHP=1,12
               PUMP(IW,MONTHP) = PUMP(IW,MONTHP) * frac_pump
            END DO
            IF(LP(5)) WRITE(75,30) IW,WFLD(IW),IXP,IYP,(PUMP(IW,K),K=1,
     $      12)
            LOCW(IW) = IXP - MINX(IYP) + 1 + ISUM(IYP)
   20    CONTINUE
      ELSE
c simulation run
         READ(18,'(I4,1x,f6.2)') NWELL,frac_pump
         BACKSPACE 18
         READ (18,'(5x,A6)') char_frac_pump
         IF (char_frac_pump .EQ. '      ') THEN
            frac_pump = 1.0
            write(*,40)
         ENDIF
         DO 50 IW = 1,NWELL
            READ(18,'(A9,2(3X,I4),12F7.2)') WFLD(IW),IXP,IYP,(PUMP
     $      (IW,MONTHP),MONTHP=1,12)
            DO MONTHP=1,12
               PUMP(IW,MONTHP) = PUMP(IW,MONTHP) * frac_pump
            END DO
            LOCW(IW) = IXP - MINX(IYP) + 1 + ISUM(IYP)
   50    CONTINUE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c      have to DO REWIND  and reREAD pumpage since they may be subject
c      to cutbacks during any given month in the succeeding years
         REWIND (18)
c
         IF(IYEAR .EQ. 1) THEN
            READ(37,'(i4)') nwell_ind_rss
            DO iw = 1,nwell_ind_rss
               READ(37,60) wfld_ind_rss(iw),ixp_ind_rss,iyp_ind_rss,
     $         (pump_ind_rss (iw,monthp),monthp=1,12)
c
               loc_ind_rss(iw) = ixp_ind_rss - minx(iyp_ind_rss) +  1+
     $         isum(iyp_ind_rss)
            END DO
         ENDIF
      ENDIF
      RETURN
   10 FORMAT(2(/),'WARNING: NO INPUT FOR MULTIPLIER OF MUNICIPAL
     $WELLFIELD PUMPAGES FOR YEAR;',2x,i5,2x,
     $'MULTIPLIER ASSUMED TO BE EQUAL TO 1.0',/)
   30 FORMAT(1X,I4,3X,A6,2I5,12F9.1)
   40 FORMAT(2(/),'WARNING: NO INPUT FOR MULTIPLIER OF MUNICIPAL
     $WELLFIELD PUMPAGES;MULTIPLIER ASSUMED TO BE EQUAL TO 1.0',/)
   60 FORMAT(A9,2(3X,I4),12F7.2)
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

northlok_res.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: northlok_res.F,v 1.5 2003/07/03 20:07:38 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/northlok_res.F,v $
C   $Name: rel-5-5 $
C
C
C  description:
C
C  SUBROUTINE to DO water budget for proposed North LOK Reservoir
C  preliminary specs are:
C   10kac @ 10ft max depth
C   Initial inflow&outflow pump capacity = 4800cfs (to be resized as necessary)
C   Location TBD, but not necessary to specifically define for SFWMM simulation
C   operations:
C    * Inflows from Lake Okeechobee to be pumped into reservoir when
C      lake stage is rising >0.25ft below the bottom pulse zone of the
C      current (aka Run25) regulation schedule.
C    * Releases back to lake when lake stage is falling > 0.5ft below the
C      bottom pulse zone.
C  source: 6/26/97 email from Cal, Subject: Restudy "Starting Point" Components
C  note: Above operating conditions have to be defined in main program.
C
C  originally written by lmb 4/29/97
C
C  input parameters:
C
C  iopt:  -1   res2lok > 0, release water to lake
C          0   no action, just DO rf-et-seepage budget
C         $1   lok2res > 0, pump water into reservoir
C  ares      = area or reservoir [acres]
C  hres_min  = minimum stage in reservoir [ft]
C  hres_max  = maximum stage in reservoir [ft]
C  hres0     = beginning of day stage [ft]
C  res_rf_day = reservoir rainfall depth [ft]
C  res_et_day = reservoir evapotranspiration IF water is available [ft]
```

```
C  qin_max   = reservoir inflow pump capacity [cfs]
C  qout_max  = reservoir outflow pump capacity [cfs]
C
C  output parameters:
C
C  actual_et = actual reservoir evapotranspiration [ft]
C  actual_seepage = actual reservoir seepage [ft]
C  res2lok   = release from reservoir to Lake Okeechobee [cfs]
C  lok2res   = inflow from Lake Okeechobee to reservoir [cfs]
C  hres      = END of day reservoir stage [ft]
C
C  other variables:
C
C  hres1     = reservoir stage after rf,et&seepage but before spill [ft]
C  hres2     = reservoir stage after spill [ft]
C  tot_res_loss = sum of actual_et and actual_seepage [acre-ft]
C
C  ***      ALL CALCULATIONS DONE IN UNITS OF CFS-DAY, UON !!!
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE northlok_res (iopt, ares, hres_min, hres_max, hres0,
     & res_rf_day, res_et_day, qin_max, qout_max, actual_et,
     & actual_seepage, res2lok, lok2res, hres, month, ifyr, iyear, iday,
     & res_seep_factor)

      INTEGER iopt

      REAL actual_et, actual_seepage, ares, hres_min, hres_max, hres,
     & hres0, lok2res, qin_max, qout_max, res_et_day, res_rf_day,
     & res2lok

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C assume reservoir properties
C  let: constant reservoir seepage loss rate = .0114 ft per day
C  Same seepage rates for proposed St. Lucie and Caloosahatchee reservoirs.

      res_seepage = .0114 * res_seep_factor

C initialize reservoir variables

      lok2res = 0.0
      res2lok = 0.0
      actual_et = 0.0
      actual_seepage = 0.0
      spill_over = 0.0
c
c update water level in reservoir based on RF-ET-Seepage
c assumption: natural processes (RF, ET & Seepage) occur
c             before any lake water enters the reservoir
c             or any reservoir water is pumpes into lake
      hres = hres0 + res_rf_day - res_et_day - res_seepage
      actual_et = res_et_day
      actual_seepage = res_seepage
```

```
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (hres .LT.0.) THEN
c calculate ET and Seepage to bring hres to zero
c assumption: ET occurs before Seepage
         h0_plus_rf = hres0 + res_rf_day
         IF (h0_plus_rf .GT. res_et_day) THEN
            actual_et = res_et_day
            actual_seepage = h0_plus_rf - actual_et
         ELSE
            actual_et = h0_plus_rf
            actual_seepage = 0.0
         ENDIF
         hres = 0.0
      ENDIF
      hres1 = hres
c
      IF (hres .GT. hres_max) THEN
         spill_over = (hres - hres_max) * ares
         hres = hres_max
c
c note: spill_over is assumed to be not subject to any structure
c       capacity constraints and is considered a loss term in
c       the reservoir water budget
      ENDIF
      hres2 = hres
      tot_res_loss = (actual_et+actual_seepage)*ares
c
      IF (iopt .EQ. -1) THEN
c make reservoir release into lake
         avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
c.....RES2LOK:
         res2lok = amin1(qout_max,avail_res_rel)
         hres = hres - res2lok * 1.9835 / ares
      ENDIF
c
      IF (iopt.EQ.1) THEN
c fill reservoir with lake water
         avail_res_sto = amax1(0.,hres_max-hres) * ares / 1.9835
c.....LOK2RES:
         lok2res = amin1(qin_max,avail_res_sto)
         hres = hres + lok2res * 1.9835 / ares
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c
c output for unit 125 in acre-ft
      IF (iopt .NE. 2) THEN
         write(125,'(i5,2i3,1x,15f10.3)') ifyr+iyear-1,month,iday ,
     $   res_rf_day*ares,spill_over ,res_et_day*ares,actual_et*ares ,
     $   res_seepage*ares,actual_seepage*ares ,tot_res_loss,res2lok
     $   * 1.9835,lok2res * 1. 9835,hres0,hres1,hres2,hres
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      ENDIF
      RETURN
      END
```

open_ascii_output_files.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: open_ascii_output_files.F,v 4.1
C                 2000/11/08 16:33:02 cwhite Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/
C                 open_ascii_output_files.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE OPENS THE REQUIRED ASCII FILES AND WRITES THE
C   APPROPRIATE HEADERS IDENTIFYING THE DATA WRITTEN TO FILES.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE open_ascii_output_files (numd, isyr, ip, xspc, yspc,
     & nlu, welldat_opt, ntotres, icol_stg, irow_stg,
     & no_pred_storms_thres_wca, itrib_hydro_cond_opt, trib_rf_et,
     & s65e_runff_wkly, no_of_small_res_budg_files, iunit_no_res_file,
     & res_file_name, ncnl_dly_slope, canal_name_dvslope, ntotasr,
     & cal_asr_flag, stl_asr_flag, charid_asr)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr3.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'loktarg.inc'
```

```
      DIMENSION charid_asr(max_n_asr,max_src_dest_pts_for_any_asr),
     & icol_stg(max_n_targ_area,max_n_targ_loc), ip(60),
     & irow_stg(max_n_targ_area,max_n_targ_loc),
     & iunit_no_res_file(max_n_sm_res), numd(12)

      CHARACTER*80 res_file_name(max_n_sm_res), tempfname
      CHARACTER*200 filename
      CHARACTER canal_name_dvslope(max_n_cnls_w_var_slpe)*5,
     & char_id_asr*6, char_id_cnl*5, charid_asr*6, fdate*24,
     & welldat_opt*5

      INTEGER cal_asr_flag, stl_asr_flag

      LOGICAL trib_check

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  OPEN REQUIRED OUTPUT FILES

C 13 Caloosahatchee basin budget components
      FILENAME=DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_c43_basin_bud.dat'
      OPEN(73,FILE = FILENAME)
c
c 16 St. Lucie basin budget components
      FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_c44_basin_bud.dat'
      OPEN(74,FILE = FILENAME)
c EAA Summary Output File
      FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_eaa_summary.dat'
      OPEN(61,FILE = FILENAME)
c Other LOSA Summary Output File
      FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_losa_other_summary.dat'
      OPEN(68,FILE = FILENAME)
c LOSA SSM Summary Output File
      FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_losa_ssm_summary.dat'
      OPEN(66,FILE = FILENAME)
c Seminole Summary Output File
      FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_tribal_summary.dat'
      OPEN(67,FILE = FILENAME)
c North Storage reservoir budget components
      IF (inorth_stor_opt .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_north_stor_bud.dat'
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         OPEN(125,FILE = FILENAME)
      ENDIF
c Taylor Creek/Nubbin Slough reservoir budget components
      IF (itcns_res_opt .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_tcnbs_stor_bud.dat'

         OPEN(126,FILE = FILENAME)
```

```
      ENDIF
      num_asr_output = ntotasr - stl_asr_flag - cal_asr_flag
c Summary of ASR Budget Components
      IF (num_asr_output .gt. 0) THEN
        FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/daily_asr_bud.dat'
        OPEN(65,FILE = FILENAME)
      ENDIF
c Summary of Water Supply Deliveries at major Structures
c
C
      IF (lp(23)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_ws_str_capac_flw.dat'
         OPEN (62, FILE = FILENAME)
      ENDIF
c
c Summary of inflows to STAs
c
C
c      FILENAME=DATADIR(1:LENGTH(DATADIR)) //
c    $          '/dly_hydraulic_det_time_stas.dat'
c        OPEN (63, FILE = FILENAME)
c
c echoof static input data for grid system
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (LP(5)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/echo_grid_statdta.dat'
         OPEN (75,FILE = FILENAME)
      ENDIF
c
c monthly canal budget summary
      IF ( LP(7)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/mthly_canal_bud.dat'
         OPEN (72,FILE = FILENAME)
      ENDIF
      IF ( LP(6) ) THEN
c yearly canal budget summary
         FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/ann_canal_bud.dat'
         OPEN (82,FILE = FILENAME)
      ENDIF
      IF       ( LP(3) .OR. LP(4) ) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/mthly_key_output.dat'
         OPEN (90,FILE = FILENAME)
      ENDIF
C end
      IF (LP(16) .OR. LP(17) .OR. LP(18)) THEN
        IF ( LP(20) ) THEN
          IF (LP(22)) THEN
            continue
          ELSE
           IF (LP(25)) THEN
            FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $      '/daily_stg_pond_et.dat'
            OPEN (70, FILE = FILENAME)
```

```
            FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $      '/daily_surface_flw.dat'
            OPEN(92, FILE = FILENAME)
           ELSE
            FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $      '/eomth_stg_pond_et.dat'
            OPEN (70, FILE = FILENAME)
            FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $      '/mthly_tot_surface_flw.dat'
            OPEN(92, FILE = FILENAME)
           ENDIF
          ENDIF
        ENDIF
      END IF
C
      FILENAME=DATADIR(1:LENGTH(DATADIR)) // '/mthly_levee_spg.dat'
      OPEN(32, FILE = FILENAME)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (lp(9)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/daily_levee_spg.dat'
         OPEN(102, FILE = FILENAME)
      ENDIF
      IF (runmode .EQ. 'SIMUL') THEN
         FILENAME=DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_minlvl_specs.dat'
         OPEN(106, FILE = FILENAME)
      ENDIF
c
      IF       (NMTR .NE. 0 ) THEN
         FILENAME= DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_stg_mon_pts.dat'
         OPEN (76, FILE = FILENAME)
      ENDIF
      IF       (NMC.NE.0.AND.LP(8)) THEN
         FILENAME= DATADIR(1:LENGTH(DATADIR)) // '/daily_canal_stg.dat'
         OPEN (78, FILE = FILENAME)
      ENDIF
c      IF       (   LP(21)  ) THEN
c         FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/daily_stg_alg_tam_trl.dat'
c         OPEN(56, FILE = FILENAME)
c      ENDIF
      IF      (   LP(11)   ) THEN
         FILENAME=DATADIR(1:LENGTH(DATADIR)) //
     $   '/ann_excess_ovlf_vol_lim.dat'
         OPEN(69, FILE = FILENAME)
      ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (LP(16)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/eomth_lok_rfetsto.dat'
         OPEN (80, FILE = FILENAME)
         WRITE(80,'(A)')'YEAR '
         WRITE(80,'(A)')'MONTH'
         WRITE(80,'(A)')'DAY  '
         WRITE(80,'(A)')'LOKMORN        LAKE OKEECHOBEE MONTHLY RAINFALL
```

```
     & (IN)'
         WRITE(80,'(A)')'RMETVOL        LAKE OKEECHOBEE MONTHLY TOTAL ET
     & VOLUME (ACRE-FEET)'
         WRITE(80,'(A)')'FINSTO_LOK     LAKE OKEECHOBEE END OF MONTH STO
     &RAGE VOLUME (ACRE-FEET)'
         WRITE(80,'(A)')'BMPLOSS        LOSS TERM IN EAA DUE TO THE EFFE
     &CTS OF BMPS'
         if (lok_split_option .EQ. 0) THEN
           WRITE(80,'(A)')'RMETVOL_RES  LAKE OKEECHOBEE MONTHLY TOTAL ET
     & VOLUME IN RESERVIOR PORTION (ACRE-FEET)'
           WRITE(80,'(A)')'FINSTO_LORES LAKE OKEECHOBEE END OF MONTH STO
     &RAGE VOLUME IN RESERVIOR PORTION (ACRE-FEET)'
         endif
         WRITE(80,'(A)')'DATA'
      ENDIF
c
c Summary of Water Supply Deliveries at major Structures
c
      IF (lp(23)) THEN
         WRITE(62,'(A)')'YEAR '
         WRITE(62,'(A)')'MONTH'
         WRITE(62,'(A)')'DAY  '
         WRITE(62,'(A)')'STRUCNAME       NAME OF STRUCTURE'
         WRITE(62,'(A)')'HEADWATER       HEADWATER'
         WRITE(62,'(A)')'TAILWATER       TAILWATER'
         WRITE(62,'(A)')'STRUCCAP        MAXIMUM CAPACITY OF STRUCTURE
     $(CFS-DAY)'
         WRITE(62,'(A)')'AVAILVOL        AVAILABLE VOLUME(CFS-DAY) OF WA
     $TER UPSTREAM FOR FLOW THRU STRUCTURE'
         WRITE(62,'(A)')'STRUCFLOW       FLOW(CFS-DAY) THRU STRUCTURE'
         WRITE(62,'(A)')'DSNEEDS         DEMANDS (CFS-DAY) DOWNSTREAM
     $OF STRUCTURE'
         WRITE(62,'(A)')'DATA'
      ENDIF
C
c
c     output stage and flow_inFORMATion for simulated weirs in WCA(s)
c
      IF (LP(12)) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/daily_weirflow.dat'
         OPEN(130, FILE = FILENAME)
      ENDIF
c
      IF(( LP(16) .OR. LP(17) .OR. LP(18)) .AND. .NOT. LP(22)) WRITE
     $(70,10) (IP(I), I = 16,18),(IP(I),I = 16,18)
      IF (LP(3) .OR. LP(4)) WRITE (90,10) IP(3),IP(4),1,1,1,0
C
C-------------------
c
c Daily summary of struc inflow into reservoir(s) recharging gw
c
      IF (NTOTRES .GT. 0) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_flw_to_res_gw.dat'
         open (131,FILE = FILENAME)
      ENDIF
c
```

```
c Daily summary of WSE LOK Operations
c
      IF (runmode .eq. 'SIMUL') THEN
       IF (iclimate_opt .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_lok_reg_est.dat'
         open (161,FILE = FILENAME)
       ENDIF
       FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $'/daily_lok_reg_wca.dat'
       open (171,FILE = FILENAME)
      ENDIF
c
c Daily LOK ET
c
      IF (LP(10)) THEN
        FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $         '/daily_lok_et.dat'
        open (181,FILE = FILENAME)
      ENDIF
c
c Daily output of Statistical component of target flow to ENP
c according to Rainfall formula computed internally(opt_ft_input = 0)
c
      IF (.NOT. opt_ft_input) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_srs_targ_flw.dat'
         open (141,FILE = FILENAME)
      ENDIF
c
c Daily summary of simulated head drops for appropriate canals
c
      IF (ncnl_dly_slope .gt. 0) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_canal_headdrop.dat'
         open (190,FILE = FILENAME)
         WRITE(190,'(A)')'YEAR '
         WRITE(190,'(A)')'MONTH'
         WRITE(190,'(A)')'DAY '
         DO I = 1,ncnl_dly_slope
           WRITE(190,'(A6)') canal_name_dvslope(i)
         ENDDO
         WRITE(190,'(A)')'DATA'
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_rf_cnl_ops_indx.dat'
         open (290,FILE = FILENAME)
         WRITE(290,'(A)')'YEAR '
         WRITE(290,'(A)')'MONTH'
         WRITE(290,'(A)')'DAY '
         DO I = 1,ncnl_dly_slope
           WRITE(290,'(A6)') canal_name_dvslope(i)
         ENDDO
         WRITE(290,'(A)')'DATA'
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C-----------------
```

```
C   POSITION FLOW, RF & PUMPAGE FILES TO CORRECT POSITION
C-----------------
      READ (36,*) ISYR_LOK
      IF (IFYR .GT. ISYR_LOK) THEN
         NMTHS_LOK = (IFYR - ISYR_LOK) * 12
         DO I = 1,NMTHS_LOK
            READ (36,*)
         ENDDO
      ENDIF
C
      READ(28,*) ISYR_PET
      NMTHS_PET = (IFYR - ISYR_PET) * 12 + (IBM - ISM)
      IF (NMTHS_PET .GT. 0) THEN
        DO 30 I = ISM,ISM+NMTHS_PET-1
          NY = (I - 1) / 12 + 1
          NUMD(2) = 28
          IF(MOD(ISYR_PET+NY-1,4) .EQ. 0) NUMD(2) = 29
          MTH = I - (NY - 1) * 12
          DO 20 J = 1,NUMD(MTH)
             READ (28,*)
   20   CONTINUE
   30   CONTINUE
      ENDIF
      READ (59,*) ISYR_RFP
      NMTHS_RFP = (IFYR - ISYR_RFP) * 12 + (IBM - ISM)
      IF (NMTHS_RFP .GT. 0) THEN
         DO 35 I = ISM,ISM+NMTHS_RFP-1
            NY = (I - 1) / 12 + 1
            NUMD(2) = 28
            IF(MOD(ISYR_RFP+NY-1,4) .EQ. 0) NUMD(2) = 29
            MTH = I - (NY - 1) * 12
            DO 25 J = 1,NUMD(MTH)
               IF(type_flow_across_ttrail .NE. 'MINDEL '.AND. .NOT. LP
     $         (24)) READ (59,*)
   25       CONTINUE
   35    CONTINUE
C-----------------
      END IF
C
      IF ( welldat_opt .EQ. 'TIMEV') THEN
c historical run
         read (18,*) ibyr_munic_wellp
         IF(IFYR .GT. ibyr_munic_wellp) THEN
            DO 50 NY = 1, IFYR - ibyr_munic_wellp
               READ (18,'(I4)') NWELL
               DO 40 J = 1,NWELL
                  READ (18,*)
   40          CONTINUE
   50       CONTINUE
         ENDIF
      ENDIF
c
c  position input file containing no of hurricane predicted
c  for each year and input file containing climate_indices(monthly)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      read (108,*) ibeg_yr_hurr_pred,no_pred_storms_thres
```

```
     $, no_pred_storms_thres_wca
      nyr_skip = ifyr - ibeg_yr_hurr_pred
      DO i = 1,nyr_skip
         read (108,*)
      END DO
c
      read (109,*) ibeg_yr_climate
      nrec_skip = ifyr - ibeg_yr_climate
      DO i = 1,nrec_skip
         read (109,*)
      END DO
c
      read (134,*) ibeg_yr_hydro
      iyear_hydro = ibeg_yr_hydro
      trib_check = .true.
 55   if (iyear_hydro .le. ifyr) then
         read (134,*) iyear_hydro,imo_hydro,iday_hydro,trib_rf_et0,
     $   s65e_runff_wkly0
         if (iyear_hydro .lt. ifyr) then
            trib_rf_et = trib_rf_et0
            s65e_runff_wkly = s65e_runff_wkly0
            trib_check = .false.
            go to 55
         endif
         if (trib_check) then
            trib_rf_et = trib_rf_et0
            s65e_runff_wkly = s65e_runff_wkly0
         endif
      endif
      backspace 134
c
      read (135,*) ibeg_yr_multi
      nrec_skip = ifyr - ibeg_yr_multi
      DO i = 1,nrec_skip
         read(135,*)
      END DO
c
c  Palmer drought severity indices
c
      read(148,*) ibeg_yr_pdsi
      nrec_skip = ifyr - ibeg_yr_pdsi
      DO i = 1,nyr_skip
         READ(148,*)
      ENDDO
c
c  daily import stages for Everglades
C
      IF (runmode .eq. 'SIMUL') THEN
      IF (wca_import_variation .EQ. 'DAILY' .OR. opt_flow_to_ts .EQ.
     $'TSRFPLN' .OR. opt_st_input) THEN
        read (83,*) ibeg_yr_stg_targ
        NMTHS_STGTRG = (IFYR - ibeg_yr_stg_targ) * 12 + (IBM - ISM)
        IF (NMTHS_STGTRG .GT. 0) THEN
          DO 57 I = ISM,ISM+NMTHS_STGTRG-1
            NY = (I - 1) / 12 + 1
            NUMD(2) = 28
            IF(MOD(ibeg_yr_stg_targ+NY-1,4) .EQ. 0) NUMD(2) = 29
```

```
            MTH = I - (NY - 1) * 12
            DO 53 J = 1,NUMD(MTH)
               READ (83,*)
   53       CONTINUE
   57     CONTINUE
        ENDIF
      ENDIF
      ENDIF
C
      DO 70 I = 1,MAXY
C-----------------
C  INITIALIZE RECHARGE TO ZERO
C-----------------
         DO 60 J = ISUM(I) + 1,ISUM (I + 1)
            ETM(J) = 0.0
   60    RCHG (J) = 0.0
   70 CONTINUE
c
C
c  determine nodal locations of import triggers for wcas
c  based on inputs of row and column
c
      IF (wca_import_variation .NE. 'DAILY') THEN
         DO isa = 1,ntotal_no_import_areas_const
            IF (nmpts_wca(isa) .GT. 0) THEN
               DO i = 1,nmpts_wca(isa)
                  node_import(isa,i) = icol_stg(isa,i) - minx(irow_stg
     $            (isa,i)) +  1+ isum(irow_stg(isa,i))
               END DO
            ENDIF
         END DO
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------
C  WRITE OUT RUN DEFINITION DATA
C-----------------
      I = IFYR
      J = IFYR+NYEAR-1
      PRINT 80, FDATE(),I,J,NINT(DT),MAXY,MAXXT,ISUM(MAXY+1), NINT
     $(XSPC),NINT(YSPC),NLU,NFLPTS,NWELL, (DETEN_DEF(K),K=2,NLU+1),(IP
     $(K),K=1,15)
C-----------------
C  WRITE MANNING'S COEFFICIENTS FOR EACH LAND USE
C-----------------
      PRINT 90,(OFML(I,1),OFML(I,2),SRZ(I),DRZ(I),I=1,NLU)
C-----------------
C  PRINT MINX & MAXX ARRAYS AND STAGE MONITOR POINTS
C-----------------
      PRINT 100, (MINX(I), I = 1,MAXY)
      PRINT 110, (MAXX(I),I = 1,MAXY)
      PRINT 120, (MXOV(I),I = 1,MAXY)
C
c
      IF (type_flow_across_ttrail .NE. 'MINDEL ') THEN
         IF (type_flow_across_ttrail .NE. 'STAGETG') THEN
            FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $      '/daily_srs_flw.dat'
```

```
            OPEN(151,FILE = FILENAME)
         ENDIF
      ENDIF
c
      IF (idbsnopt1 .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c43_basin_res.dat'
         open(87,file = filename)
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c43_res_bud.dat'
         open(89,file = filename)
      ENDIF
      IF (idbsnopt2 .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c44_basin_res.dat'
         open(88,file = filename)
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c44_res_bud.dat'
         open(91,file = filename)
      ENDIF
      IF (cal_asr_flag .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c43_asr_bud.dat'
         open(84,file = filename)
      ENDIF
      IF (stl_asr_flag .EQ. 1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_c44_asr_bud.dat'
         open(86,file = filename)
      ENDIF
c
c  Header for Caloosahatchee Basin water budget
c
      WRITE(73,'(A)')'YEAR'
      WRITE(73,'(A)')'MONTH'
      WRITE(73,'(A)')'DAY'
      WRITE(73,'(A)')'C43DMD         C43 BASIN SUPPLEMENTAL DEMAND, WITH
     $OUT DISTON (AC-FT)'
      WRITE(73,'(A)')'C43DMDREV      C43 BASIN SUPPLEMENTAL DEMAND, INCL
     $UDING EFFECTS FROM DISTON (AC-FT)'
      WRITE(73,'(A)')'C43DMDLOK      C43 BASIN SUPPLEMENTAL DEMAND ON LO
     $K (AC-FT)'
      WRITE(73,'(A)')'C43BSALLOC     C43 BASIN ALLOCATION ACCORDING TO S
     $SM'
      WRITE(73,'(A)')'C43DMBYLOK     C43 BASIN DEMAND MET MY LOK (AC-FT)
     $'
      WRITE(73,'(A)')'C43BRUN        C43 BASIN RUNOFF, WITHOUT DISTON (A
     $C-FT)'
      WRITE(73,'(A)')'C43BRUNREV     C43 BASIN RUNOFF, INCLUDING EFFECTS
     $ FROM DISTON (AC-FT)'
      WRITE(73,'(A)')'C43TEXCRUN     C43 BASIN TOTAL EXCESS RUNOFF (AC-F
     $T)'
      WRITE(73,'(A)')'C43BRFTLOK     C43 BASIN RUNOFF TO LOK (AC-FT)'
      WRITE(73,'(A)')'C43BRFTEST     C43 BASIN RUNOFF TO ESTUARY THRU S7
     $9'
      WRITE(73,'(A)')'STGLOK         LOK STAGE (FT NGVD)'
      WRITE(73,'(A)')'CALESTDMD      CALOOSAHATCHEE ESTUARY DEMAND'
```

```
      WRITE(73,'(A)')'LKSDCALES0     MEAN DAILY LOK SUPPLEMENTAL DEMAND
     $FOR CURRENT MOMTH'
      WRITE(73,'(A)')'LKSDCALEST     LOK SUPPLEMENTAL DEMAND FOR CAL EST
     $UARY FOR CURR TIME STEP'
      WRITE(73,'(A)')'C43BKFCAP      MAXIMUM CAPACITY FOR BACKFLOW OF CA
     $LOOS RUNOFF TO LOK'
      WRITE(73,'(A)')'S77NONREG      TOTAL FLOW ,OTHER THAN REGULATORY,T
     $HRU S77'
      WRITE(73,'(A)')'S79NONREG      TOTAL FLOW ,OTHER THAN REGULATORY,T
     $HRU S79'
      WRITE(73,'(A)')'C43BRFEST      C43 BASIN RUNOFF USED TO MEET ESTUA
     $RINE DEMANDS'
      WRITE(73,'(A)')'DATA'
c
c   Header for St Lucie Basin Water Budget
c
      WRITE(74,'(A)')'YEAR'
      WRITE(74,'(A)')'MONTH'
      WRITE(74,'(A)')'DAY'
      WRITE(74,'(A)')'C44DMD         C44 BASIN SUPPLEMENTAL DEMAND (AC-F
     $T)'
      WRITE(74,'(A)')'C44DMDLOK      C44 BASIN SUPPLEMENTAL DEMAND ON LO
     $K (AC-FT)'
      WRITE(74,'(A)')'C44BSALLOC     C44 BASIN ALLOCATION ACCORDING TO S
     $SM (AC-FT)'
      WRITE(74,'(A)')'C44DMBYLOK     C44 BASIN DEMAND MET MY LOK (AC-FT)
     $'
      WRITE(74,'(A)')'C44BRUNOFF     C44 BASIN RUNOFF (AC-FT)'
      WRITE(74,'(A)')'C44TEXCRUN     C44 BASIN TOTAL EXCESS RUNOFF (AC-F
     $T)'
      WRITE(74,'(A)')'C44BRFTLOK     C44 BASIN RUNOFF TO LOK (AC-FT)'
      WRITE(74,'(A)')'C44BRFTEST     C44 BASIN RUNOFF TO ESTUARY THRU S8
     $0'
      WRITE(74,'(A)')'STGLOK         LOK STAGE (FT NGVD)'
      WRITE(74,'(A)')'STLESTDMD      ST LUCIE ESTUARY DEMAND'
      WRITE(74,'(A)')'STLEXCINF      ST LUCIE TRIBUTARY INFLOW IN EXCESS
     $OF ESTUARINE DEMANDS'
      WRITE(74,'(A)')'STLINF         ST LUCIE TRIBUTARY INFLOW'
      WRITE(74,'(A)')'LKSDSTLES0     MEAN DAILY LOK SUPPLEMENTAL DEMAND
     $FOR CURRENT MONTH'
      WRITE(74,'(A)')'LKSDSTLEST     LOK SUPPLEMENTAL DEMAND FOR STL EST
     $UARY FOR CURR TIME STEP'
      WRITE(74,'(A)')'C44BKFCAP      MAXIMUM CAPACITY FOR BACKFLOW OF ST
     $ LUCIE BASIN RUNOFF TO LOK'
      WRITE(74,'(A)')'S308NONREG     TOTAL FLOW ,OTHER THAN REGULATORY,T
     $HRU S308'
      WRITE(74,'(A)')'S80NONREG      TOTAL FLOW ,OTHER THAN REGULATORY,T
     $HRU S80'
      WRITE(74,'(A)')'C44BRFEST      C44 BASIN RUNOFF USED TO MEET ESTUA
     $RINE DEMANDS'
      WRITE(74,'(A)')'DATA'
c
c     Header for Caloos Basin/Reservior Components
c
      IF (idbsnopt1 .EQ. 1) THEN
      WRITE(87,'(A)')'YEAR'
      WRITE(87,'(A)')'MONTH'
```

```
      WRITE(87,'(A)')'DAY'
      WRITE(87,'(A)')'SPILLOVER      SPILLOVER (OVERFLOW) FROM RESERVOIR
     $ NOT MEETING ESTUARINE REQUIREMENTS'
      WRITE(87,'(A)')'C43BRUNFF      C43 BASIN RUNOFF (INCLUDING EFFECTS
     $ FROM DISTON)'
      WRITE(87,'(A)')'C43BSDMND      C43 BASIN REMAINING SUPP ESTUARINE
     $DEMANDS'
      WRITE(87,'(A)')'C43BTEST       C43 BASIN RUNOFF TO MEET ESTUARINE
     $DEMANDS'
      WRITE(87,'(A)')'C43BTRES       C43 BASIN RUNOFF TO PROP RESERVOIR'
      WRITE(87,'(A)')'ROTLKOEST      C43 BASIN RUNOFF TO LOK OR ESTUARY'
      WRITE(87,'(A)')'C43BDMND       C43 BASIN DEMAND'
      WRITE(87,'(A)')'C43UNMDMD      C43 BASIN UNMET DEMAND'
      WRITE(87,'(A)')'RESTC43B       PROP RESERVOIR FLOW TO MEET C43 BAS
     $IN DEMANDS'
      WRITE(87,'(A)')'RESTEST        PROP RESERVOIR FLOW TO MEET ESTUARI
     $NE DEMANDS'
      WRITE(87,'(A)')'UNMESTDMD      UNMET ESTUARINE DEMANDS'
      WRITE(87,'(A)')'SPILLESTD      SPILLOVER THAT MEETS ESTUARINE DEMA
     $NDS'
      WRITE(87,'(A)')'RESSTGBEG      RESERVOIR WATER DEPTH AT BEG OF TIM
     $E STEP'
      WRITE(87,'(A)')'RESSTG         CURRENT RESERVOIR WATER DEPTH'
      WRITE(87,'(A)')'DELRESSTO      CHANGE IN RESERVOIR STORAGE FROM PR
     $EV DAY'
      WRITE(87,'(A)')'RESTASR        INJECTION OF RESERVOIR WATER TO ASR
     $'
      WRITE(87,'(A)')'DATA'
      ENDIF
c
c     Header for St Lucie Basin/Reservior Components
c
      IF (idbsnopt2 .EQ. 1) THEN
      WRITE(88,'(A)')'YEAR'
      WRITE(88,'(A)')'MONTH'
      WRITE(88,'(A)')'DAY'
      WRITE(88,'(A)')'SPILLOVER      SPILLOVER (OVERFLOW) FROM RESERVOIR
     $ NOT MEETING ESTUARINE REQUIREMENTS'
      WRITE(88,'(A)')'C44BRUNFF      C44 BASIN RUNOFF'
      WRITE(88,'(A)')'C44BSDMND      C44 BASIN REMAINING SUPP ESTUARINE
     $DEMANDS'
      WRITE(88,'(A)')'C44BTEST       C44 BASIN RUNOFF TO MEET ESTUARINE
     $DEMANDS'
      WRITE(88,'(A)')'C44BTRES       C44 BASIN RUNOFF TO PROP RESERVOIR'
      WRITE(88,'(A)')'ROTLKOEST      C44 BASIN RUNOFF TO LOK OR ESTUARY'
      WRITE(88,'(A)')'C44BDMND       C44 BASIN DEMAND'
      WRITE(88,'(A)')'C44UNMDMD      C44 BASIN UNMET DEMAND'
      WRITE(88,'(A)')'RESTC44B       PROP RESERVOIR FLOW TO MEET C44 BAS
     $IN DEMANDS'
      WRITE(88,'(A)')'RESTEST        PROP RESERVOIR FLOW TO MEET ESTUARI
     $NE DEMANDS'
      WRITE(88,'(A)')'UNMESTDMD      UNMET ESTUARINE DEMANDS'
      WRITE(88,'(A)')'SPILLESTD      SPILLOVER THAT MEETS ESTUARINE DEMA
     $NDS'
      WRITE(88,'(A)')'RESSTGBEG      RESERVOIR WATER DEPTH AT BEG OF TIM
     $E STEP'
      WRITE(88,'(A)')'RESSTG         CURRENT RESERVOIR WATER DEPTH'
```

```
      WRITE(88,'(A)')'DELRESSTO      CHANGE IN RESERVOIR STORAGE FROM PR
     $EV DAY'
      WRITE(88,'(A)')'RESTASR        INJECTION OF RESERVOIR WATER TO ASR
     $'
      WRITE(88,'(A)')'DATA'
      ENDIF
C
c     Header for Caloosahatchee Reservoir Budget
c
      IF (idbsnopt1 .EQ. 1) THEN
      WRITE(89,'(A)')'YEAR'
      WRITE(89,'(A)')'MONTH'
      WRITE(89,'(A)')'DAY'
      WRITE(89,'(A)')'RAINFALL       RAINFALL(ACRE-FT)'
      WRITE(89,'(A)')'PET            POTENTIAL EVAPORATION (ACRE-FT)'
      WRITE(89,'(A)')'ACTUALET       ACTUAL ET LOSS (ACRE-FT)'
      WRITE(89,'(A)')'POTSEEP        POTENTIAL SEEPAGE LOSS WHEN RESERVO
     $IR HAS SUFFICIENT WATER'
      WRITE(89,'(A)')'ACTUALSP       ACTUAL SEEPAGE LOSS FROM RESERVOIR(
     $ACRE-FT)'
      WRITE(89,'(A)')'TOTLOSS        TOTAL RESERV LOSS(ET PLUS SEEPAGE)'
      WRITE(89,'(A)')'BEGDEPTH       RESERVOIR WATER DEPTH (FT) AT BEGIN
     $NING OF DAY'
      WRITE(89,'(A)')'CURRDPTH       CURRENT RESERVOIR WATER DEPTH (FT)'
      WRITE(89,'(A)')'DATA'
      ENDIF
C
c     Header for St Lucie Reservoir Budget
c
      IF (idbsnopt2 .EQ. 1) THEN
      WRITE(91,'(A)')'YEAR'
      WRITE(91,'(A)')'MONTH'
      WRITE(91,'(A)')'DAY'
      WRITE(91,'(A)')'RAINFALL       RAINFALL(ACRE-FT)'
      WRITE(91,'(A)')'PET            POTENTIAL EVAPORATION (ACRE-FT)'
      WRITE(91,'(A)')'ACTUALET       ACTUAL ET LOSS (ACRE-FT)'
      WRITE(91,'(A)')'POTSEEP        POTENTIAL SEEPAGE LOSS WHEN RESERVO
     $IR HAS SUFFICIENT WATER'
      WRITE(91,'(A)')'ACTUALSP       ACTUAL SEEPAGE LOSS FROM RESERVOIR(
     $ACRE-FT)'
      WRITE(91,'(A)')'TOTLOSS        TOTAL RESERV LOSS(ET PLUS SEEPAGE)'
      WRITE(91,'(A)')'BEGDEPTH       RESERVOIR WATER SEPTH (FT) AT BEGIN
     $NING OF DAY'
      WRITE(91,'(A)')'CURRDPTH       CURRENT RESERVOIR WATER DEPTH (FT)'
      WRITE(91,'(A)')'DATA'
      ENDIF
C
c     Header for Caloosahatchee ASR Budget
c
      IF (cal_asr_flag .EQ. 1) THEN
      WRITE(84,'(A)')'YEAR'
      WRITE(84,'(A)')'MONTH'
      WRITE(84,'(A)')'DAY'
      WRITE(84,'(A)')'ASRVOL0        ASR BUBBLE SIZE AT BEGINNING OF TIM
     $E STEP'
      WRITE(84,'(A)')'ASRVOL         CURRENT ASR BUBBLE SIZE'
      WRITE(84,'(A)')'ASRINJLOSS     ASR INJECTION LOSS (ASSUMED 30% OF
```

```
     $VOLUME OF WATER INJECTED'
      WRITE(84,'(A)')'ASRSTOLOSS     ASR BUBBLE STORAGE LOSS AS FUNCTION
     $ OF DECAY RATE (ASSUMED ZERO)'
      WRITE(84,'(A)')'RESTASR        CALOOS RESERVOIR WATER INJECTED INT
     $O ASR'
      WRITE(84,'(A)')'ASRTBASIN      RECOVERY OF ASR WATER TO MEET BASIN
     $ DEMANDS'
      WRITE(84,'(A)')'ASRTEST        RECOVERY OF ASR WATER TO MEET REMAI
     $NING ESTUARINE DEMANDS'
      WRITE(84,'(A)')'DATA'
      ENDIF
C
c     Header for St Lucie ASR Budget
c
      IF (stl_asr_flag .EQ. 1) THEN
      WRITE(86,'(A)')'YEAR'
      WRITE(86,'(A)')'MONTH'
      WRITE(86,'(A)')'DAY'
      WRITE(86,'(A)')'ASRVOL0        ASR BUBBLE SIZE AT BEGINNING OF TIM
     $E STEP'
      WRITE(86,'(A)')'ASRVOL         CURRENT ASR BUBBLE SIZE'
      WRITE(86,'(A)')'ASRINJLOSS     ASR INJECTION LOSS (ASSUMED 30% OF
     $VOLUME OF WATER INJECTED'
      WRITE(86,'(A)')'ASRSTOLOSS     ASR BUBBLE STORAGE LOSS AS FUNCTION
     $ OF DECAY RATE (ASSUMED ZERO)'
      WRITE(86,'(A)')'RESTASR        ST LUCIE RESERVOIR WATER INJECTED I
     $NTO ASR'
      WRITE(86,'(A)')'ASRTBASIN      RECOVERY OF ASR WATER TO MEET BASIN
     $ DEMANDS'
      WRITE(86,'(A)')'ASRTEST        RECOVERY OF ASR WATER TO MEET REMAI
     $NING ESTUARINE DEMANDS'
      WRITE(86,'(A)')'DATA'
      ENDIF
c
c  OPEN output files for daily water budgets for small reservoirs
c
      DO i = 1,no_of_small_res_budg_files
        tempfname = res_file_name(i)
        ibeg = iscan(tempfname,80,-80,'/',1,1,ipos2)
        FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/' // tempfname(ibeg
     $  + 1:LENGTH(tempfname))
        OPEN(iunit_no_res_file(i),file = filename)
      ENDDO
c
c  WRITE header on reservoir output budget files
c
       DO i = 1,no_of_small_res_budg_files
          WRITE(iunit_no_res_file(i),'(A)')'YEAR '
          WRITE(iunit_no_res_file(i),'(A)')'MONTH'
          WRITE(iunit_no_res_file(i),'(A)')'DAY '
          WRITE(iunit_no_res_file(i),'(A)')'RESSTAGE       RESERVOIR STA
     $GE'
          WRITE(iunit_no_res_file(i),'(A)')'CELLSTAGE      CELL STAGE OU
     $TSIDE RESERV AT KEY LOCATION'
          WRITE(iunit_no_res_file(i),'(A)')'RAINFALL       RAINFALL(ACRE
     $_FT)'
          WRITE(iunit_no_res_file(i),'(A)')'RECFASR        RECOVERY FROM
```

```
     $ ASR(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'STRINFLOW      STRUCTURAL IN
     $FLOW(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'ET             EVAPOTRANSPIR
     $ATION(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'SEEPTGW        SEEPAGE TO LO
     $CAL GROUNDWATER(ACRE-FT) OR ELSEWHERE'
          WRITE(iunit_no_res_file(i),'(A)')'STROUTFLW      STRUCTURAL OU
     $TFLOW(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'LVSEEP         LEVEE SEEPAGE
     $ LOSS ACROSS EVERGLADES/LEC BARRIER(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'INJTASR        INJECTION OF
     $WATER TO ASR(ACRE-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'RESID          RESIDUAL(ACRE
     $-FT)'
          WRITE(iunit_no_res_file(i),'(A)')'DATA'
       ENDDO
c
c  WRITE header on output file for min flows and levels
c
      IF (runmode .EQ. 'SIMUL') THEN
         WRITE(106,'(A)')'YEAR '
         WRITE(106,'(A)')'MONTH'
         WRITE(106,'(A)')'DAY '
         WRITE(106,'(A)')'LOKCRIT        LOK STAGE BELOW CRITICAL LEVEL
     $FOR LOK WS RELEASES TO WCA1(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA1CANAL      CANAL STAGE IN WCA1 AT OR BELOW
     $ FLOOR(1-YES,0-NO)'
         DO i = 1,no_of_subareas_floor(1)
            WRITE(106,130) (gage_name(1,i,j),j=1, nmpts_wca_floor(1,i))
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         END DO
         WRITE(106,'(A)')'WCA1MAR        MARSH CRITERIA MET FOR WS DELIV
     $ERY FROM LOK TO SA1 VIA WCA1(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA1LOKDMD     DEMAND IN SA1 EXISTS FOR LOK(1-
     $YES,0-NO)'
         WRITE(106,'(A)')'WCA1LOKDEL     DELIVERY MADE FROM LOK TO SA1 V
     $IA WCA1(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA2ACANAL     CANAL STAGE IN WCA2A AT OR BELO
     $W FLOOR(1-YES,0-NO)'
         DO i = 1,no_of_subareas_floor(2)
            WRITE(106,131) (gage_name(2,i,j),j=1, nmpts_wca_floor(2,i))
         END DO
         WRITE(106,'(A)')'WCA2AMAR       MARSH CRITERIA MET FOR WS DELIV
     $ERY FROM LOK TO SA2 VIA WCA2A(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA2LOKDMND    DEMAND IN SA2 EXISTS FOR LOK(1-
     $YES,0-NO)'
         WRITE(106,'(A)')'WCA2LOKDEL     DELIVERY MADE FROM LOK TO SA2 V
     $IA WCA2A(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA3ACANAL     CANAL STAGE IN WCA3A AT OR BELO
     $W FLOOR(1-YES,0-NO)'
         DO i = 1,no_of_subareas_floor(4)
            WRITE(106,132) (gage_name(4,i,j),j=1, nmpts_wca_floor(4,i))
         END DO
         WRITE(106,'(A)')'WCA3AMAR       MARSH CRITERIA MET FOR WS DELIV
     $ERY FROM LOK TO SA3 VIA WCA3A(1-YES,0-NO)'
         WRITE(106,'(A)')'WCA3LOKDMD     DEMAND IN SA3 EXISTS FOR LOK(1-
```

```
     $YES,0-NO)'
         WRITE(106,'(A)')'WCA3LOKDEL     DELIVERY MADE FROM LOK TO SA3 V
     $IA WCA3A(1-YES,0-NO)'
         WRITE(106,'(A)')'DATA'
c
c  print header for daily WSE LOK operations summary
      IF (iclimate_opt .EQ. 1 .AND. itrib_hydro_cond_opt .EQ. 1) THEN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         WRITE(161,'(A)')'YEAR '
         WRITE(161,'(A)')'MONTH'
         WRITE(161,'(A)')'DAY '
         WRITE(161,'(A)')'LOKZONE        ZONE OF OPER SCHEDULE LOK STAGE
     $ IS IN'
         WRITE(161,'(A)')'THYDROCOND     CLASSIFICATION OF LOK TRIB HYDR
     $O COND(RAIN AND S-65E INFLOW)'
         WRITE(161,'(A)')'LOKSTGCOND     CLASSIFICATION OF LAKE STG COND
     $'
         WRITE(161,'(A)')'SEASFOREC      CLASSIFICATION OF SEASONAL (6-M
     $ONTH FORECAST OF LOK INFLOW'
         WRITE(161,'(A)')'MSEASFOREC     CLASSIFICATION OF MULTI-SEASONA
     $L FORECAST OF LOK INFLOW'
         WRITE(161,'(A)')'S77REG         LOK REG RELEASES THRU S77(CFS-D
     $AY'
         WRITE(161,'(A)')'S79FLOW        TOTAL FLOW THRU S-79(CFS-DAY)'
         WRITE(161,'(A)')'S308REG        LOK REG RELEASES THRU S308(CFS-
     $DAY)'
         WRITE(161,'(A)')'S80FLOW        TOTAL FLOW THRU S80(CFS-DAY)'
         WRITE(161,'(A)')'DATA'
c
         WRITE(171,'(A)')'YEAR '
         WRITE(171,'(A)')'MONTH'
         WRITE(171,'(A)')'DAY '
         WRITE(171,'(A)')'LOKZONE        ZONE OF OPER SCHEDULE LOK STAGE
     $ IS IN'
         WRITE(171,'(A)')'THYDROCOND     CLASSIFICATION OF LOK TRIB HYDR
     $O COND(RAIN AND S-65E INFLOW)'
         WRITE(171,'(A)')'SEASFOREC      CLASSIFICATION OF SEASONAL (6-M
     $ONTH FORECAST OF LOK INFLOW)'
         WRITE(171,'(A)')'MSEASFOREC     CLASSIFICATION OF MULTI-SEASONA
     $L FORECAST OF LOK INFLOW'
         WRITE(171,'(A)')'ENEEDSMIAC     EVERGLADES NEEDS IN WCA MIAMIC
     $MEETS(YES OR NO)'
         WRITE(171,'(A)')'ENEEDSNNRC     EVERGLADES NEEDS IN WCA NNRC ME
     $ETS(YES OR NO)'
         WRITE(171,'(A)')'ENNEDSWPBC     EVERGLADES NEEDS IN WCA WPBC ME
     $ETS(YES OR NO)'
         WRITE(171,'(A)')'ENEEDSHLSB     EVERGLADES NEEDS IN WCA HLSBC M
     $EETS(YES OR NO)'
         WRITE(171,'(A)')'S354REG        LOK REG RELEASES THRU S354'
         WRITE(171,'(A)')'NNRCREG        LOK REG RELEASES VIA NNRC IN EA
     $A'
         WRITE(171,'(A)')'S352REG        LOK REG RELEASES THRU S352'
         WRITE(171,'(A)')'HLSBREG        LOK REG RELEASES VIA HLSB C IN
     $EAA'
         WRITE(171,'(A)')'LREGMIAC       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA MIAMIC'
         WRITE(171,'(A)')'LREGNNRC       CLASSIFICATION OF LIMIT TO LOK
```

```
     $REG RELEASES VIA NNRC'
         WRITE(171,'(A)')'LREGWPBC       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA WPB C'
         WRITE(171,'(A)')'LREGHLSB       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA HLSB C'
         WRITE(171,'(A)')'DATA'
      ENDIF
      IF (iclimate_opt .NE. 1) THEN
         WRITE(171,'(A)')'YEAR '
         WRITE(171,'(A)')'MONTH'
         WRITE(171,'(A)')'DAY '
         WRITE(171,'(A)')'LOKZONE        ZONE OF OPER SCHEDULE LOK STAGE
     $ IS IN'
         WRITE(171,'(A)')'ENEEDSMIAC     EVERGLADES NEEDS IN WCA MIAMIC
     $MEETS(YES OR NO)'
         WRITE(171,'(A)')'ENEEDSNNRC     EVERGLADES NEEDS IN WCA NNRC ME
     $ETS(YES OR NO)'
         WRITE(171,'(A)')'ENNEDSWPBC     EVERGLADES NEEDS IN WCA WPBC ME
     $ETS(YES OR NO)'
         WRITE(171,'(A)')'ENEEDSHLSB     EVERGLADES NEEDS IN WCA HLSBC M
     $EETS(YES OR NO)'
         WRITE(171,'(A)')'LREGMIAC       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA MIAMIC'
         WRITE(171,'(A)')'LREGNNRC       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA NNRC'
         WRITE(171,'(A)')'LREGWPBC       CLASSIFICATION OF LIMIT TO LOK
     $REG RELEASES VIA WPB C'
         WRITE(171,'(A)')'LREGHLSB       CLASSIFICATION OF LIMIT TO LOK
     $REGRELEASES VIA HLSB C'
         WRITE(171,'(A)')'S354REG        LOK REG RELEASES THRU S354'
         WRITE(171,'(A)')'NNRCREG        LOK REG RELEASES VIA NNRC IN EA
     $A'
         WRITE(171,'(A)')'S352REG        LOK REG RELEASES THRU S352'
         WRITE(171,'(A)')'HLSBREG        LOK REG RELEASES VIA HLSB C IN
     $EAA'
         WRITE(171,'(A)')'DATA'
      ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  WRITE header to output file of unit no 66
c
      WRITE(66,'(A)')'YEAR '
      WRITE(66,'(A)')'MONTH'
      WRITE(66,'(A)')'DAY '
      WRITE(66,'(A)')'TSDMDLOSA      TOTAL SUPP DMND IN LOSA TO BE MET B
     $Y LOK (AC-FT)'
      WRITE(66,'(A)')'LOKSSMINDX     INDEX INDICATING WHETHER LOK IS IN
     $SSM(3 -YES,2-WARNING ZONE,1-WATCH ZONE,0-NO ZONE)'
      WRITE(66,'(A)')'LKALLOCSSM     ALLOCATION FROM LOK TO LOSA ACCORDI
     $NG TO SSM (AC-FT)'
      WRITE(66,'(A)')'SSMEXCESS      SSM ALLOCATION FROM LOK TO LOSA IN
     $EXCESS OF ACTUAL DMND (AC-FT)'
      WRITE(66,'(A)')'SSMCUTBACK     TOTAL CUTBACK VOLUME FOR ALL LOSA A
     $CCORDING TO SSM (AC-FT)'
      WRITE(66,'(A)')'CUTBACK        TOTAL CUTBACK VOLUME FOR ALL LOSA A
     $FTER MINIMUM DELIVERY (AC-FT)'
```

```
      WRITE(66,'(A)')'SSMFRACMET     FRACTION OF LOSA SUPP DMND TO BE ME
     $T ACCORDING TO SSM'
      WRITE(66,'(A)')'SLOSA_MET      SUPP DMND IN LOSA MET BY LOK (AC-FT
     $)'
      WRITE(66,'(A)')'SSMFRACMET0    FRACTION OF LOSA SUPP DMND TO BE ME
     $T ACCORDING TO SSM WITHOUT RULE 40E-21'
      WRITE(66,'(A)')'IPHASE         ASSUMED WATER SHORTAGE PHASE BASED
     $ON MODEL IMPLEMENTATION OF RULE 40E-21'
      WRITE(66,'(A)')'DATA '

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  WRITE header to output file of unit no 61
c
c EAA by definition includes Miami, NNRHill, WPB basins and Sugar ranch
c and excludes the 298 districts.

      WRITE(61,'(A)')'YEAR '
      WRITE(61,'(A)')'MONTH'
      WRITE(61,'(A)')'DAY '
      WRITE(61,'(A)')'TDMDEAA        TOTAL EAA CROP REQUIREMENTS (AC-FT)
     $'
      WRITE(61,'(A)')'TDMDMBYRF      EAA DMND MET BY RAIN (AC-FT)'
      WRITE(61,'(A)')'TDMDMBLSTO     EAA DMND MET BY LOCAL STORAGE (AC-F
     $T)'
      WRITE(61,'(A)')'TSDMDEAA       TOTAL SUPP DMND IN EAA (AC-FT)'
c      WRITE(61,'(A)')'WSEAAR_EAA     EAA DMND MET BY EAA RESERVOIR (AC-F
c     $T)'
c      WRITE(61,'(A)')'WSASR_EAA      EAA DMD MET BY ASR (AC-FT)'
      WRITE(61,'(A)')'SDMDLOKEAA     TOTAL SUPP DMND IN EAA TO BE MET BY
     $ LOK (AC-FT)'
      WRITE(61,'(A)')'SDMDLKMIA      SUPP DMND IN MIAMI BASIN (INCLUDING
     $ SUGAR RANCH) TO BE MET BY LOK (AC-FT)'
      WRITE(61,'(A)')'SDMDLKNNRH     SUPP DMND IN NNRHIL BASIN TO BE MET
     $ BY LOK (AC-FT)'
      WRITE(61,'(A)')'SDMDLKWPB      SUPP DMND IN WPB CANAL BASIN TO BE
     $MET BY LOK (AC-FT)'
      WRITE(61,'(A)')'SSUGDMND       SUPP DMND IN SUGAR RANCH PLANTATION
     $ (AC-FT)'
      WRITE(61,'(A)')'TWSLOKEAA      TOTAL WS DELIV MADE TO EAA FROM LOK
     $ (AC-FT)'
      WRITE(61,'(A)')'WSLOKMIAB      SUPP WS DELIV TO MIAMI CANAL BASIN
     $(INCLUDING SUGAR RANCH) FROM LOK (AC-FT)'
      WRITE(61,'(A)')'WSLOKNNRC      SUPP WS DELIV TO NNR CANAL BASIN FR
     $OM LOK (AC-FT)'
      WRITE(61,'(A)')'WSLOKHILL      SUPP WS DELIV TO HILL CANAL BASIN F
     $ROM LOK (AC-FT)'
      WRITE(61,'(A)')'WSLOKWPBB      SUPP WS DELIV TO WPB CANAL BASIN FR
     $OM LOK (AC-FT)'
      WRITE(61,'(A)')'MIAREMAIN      TOTAL EXCESS WATER REMAINING IN MIA
     $MI CANAL BASIN IN EAA (AC-FT)'
      WRITE(61,'(A)')'NNRHREMAIN     TOTAL EXCESS WATER REMAINING IN NNR
     $-HIL CANAL BASIN IN EAA (AC-FT)'
      WRITE(61,'(A)')'WPBREMAIN      TOTAL EXCESS WATER REMAINING IN WPB
     $ CANAL BASIN IN EAA (AC-FT)'
      WRITE(61,'(A)')'WPBTHILDIV     DIVERSION OF WPB CANAL BASIN RUNOFF
     $ TO HILL BASIN (AC-FT)'
```

```
      WRITE(61,'(A)')'EXCWSOMIA      EXCESS WATER GOING SOUTH FROM MIAMI
     $ CANAL BASIN (AC-FT)'
      WRITE(61,'(A)')'EXCWSONNR      EXCESS WATER GOING SOUTH FROM NNR C
     $ANAL BASIN (AC-FT)'
      WRITE(61,'(A)')'EXCWSOHILL     EXCESS WATER GOING SOUTH FROM HILL
     $CANAL BASIN (AC-FT)'
      WRITE(61,'(A)')'EXCWSOWPB      EXCESS WATER FROM WPB CANAL BASIN (
     $AC-FT)'
      WRITE(61,'(A)')'ETREDEAA       TOTAL ET REDUCTION IN EAA'
      WRITE(61,'(A)')'MIAFORPMP      INDEX FOR FOREPUMPING THRU S-354(1-
     $YES,0-NO)'
      WRITE(61,'(A)')'NNRHFORPMP     INDEX FOR FOREPUMPING THRU S-351(1-
     $YES,0-NO)'
      WRITE(61,'(A)')'WPBFORPMP      INDEX FOR FOREPUMPING THRU S-352(1-
     $YES,0-NO)'
      WRITE(61,'(A)')'DATA'

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  WRITE header to output file of unit no 68
c
      WRITE(68,'(A)')'YEAR '
      WRITE(68,'(A)')'MONTH'
      WRITE(68,'(A)')'DAY '
      WRITE(68,'(A)')'S4DISTDMD         S-4 DISTON SUPP DMD (AC-FT)'
      WRITE(68,'(A)')'S4OTHERDMD        OTHER S-4 SUPP DMD (AC-FT)'
      WRITE(68,'(A)')'S4DMD             OVERALL S-4 BASIN SUPP DMD INCLU
     $DING EFFECTS FROM DISTON (AC-FT)'
      WRITE(68,'(A)')'S4DMDMET          S4 BASIN SUPP DMD MET BY LOK (AC
     $-FT)'
      WRITE(68,'(A)')'S4DISTRUN         S-4 DISTON RUNOFF (AC-FT)'
      WRITE(68,'(A)')'S4OTHERRUN        OTHER S-4 RUNOFF (AC-FT)'
      WRITE(68,'(A)')'S4RUN             OVERALL S-4 BASIN RUNOFF INCLUDI
     $NG EFFECTS FROM DISTON (AC-FT)'
      WRITE(68,'(A)')'S235TC            S4 BASIN RUNOFF TO CALOOS BASIN
     $VIA S235 (AC-FT)'
      WRITE(68,'(A)')'S4BSNTLK          S4 BASIN RUNOFF TO LOK VIA S4 PU
     $MP AND DRAINAGE CULVERTS (AC-FT)'
      WRITE(68,'(A)')'S236DMD           S236 BASIN SUPP DMD (AC-FT)'
      WRITE(68,'(A)')'S236DMDMET        S236 BASIN SUPP DMD MET BY LOK (
     $AC-FT)'
      WRITE(68,'(A)')'S236RUN           S236 BASIN RUNOFF (AC-FT)'
      WRITE(68,'(A)')'298DDMD           298 DISTRICTS SUPP DMND (AC-FT)'
      WRITE(68,'(A)')'298DDMDLOK        298 DISTRICTS SUPP DMND MET BY L
     $OK (AC-FT)'
      WRITE(68,'(A)')'RUNOFF298D        RUNOFF FROM 298 DISTRICTS (AC-FT
     $)'
      WRITE(68,'(A)')'L8DMD             L-8 BASIN SUPP DMND (AC-FT)'
      WRITE(68,'(A)')'L8DMDMET          L-8 BASIN SUPP DMD MET BY LOK (A
     $C-FT)'
      WRITE(68,'(A)')'L8 RUNOFF         L-8 BASIN RUNOFF (AC-FT)'
      WRITE(68,'(A)')'ISTKDMD           LOWER ISTOKPOGA BASIN SUPP DMD (
     $AC-FT)'
      WRITE(68,'(A)')'ISTACCUM          INDEX INDICATING LOWER ISTOKPOGA
     $ ACCUMULATION OF DEMAND DUE TO CONVEYANCE LIMITS THROUGH G207 AND
     $G208(1=YES,0=NO)'
      WRITE(68,'(A)')'ISTKACCDMD        ACCUMULATED SUPP DMD VOLUME TO B
```

```
     $E CARRIED OVER TO NEXT DAY (AC-FT)'
      WRITE(68,'(A)')'ISTKDMDMET        LOWER ISTOKPOGA BASIN SUPP DMD M
     $ET BY LOK (AC-FT)'
      WRITE(68,'(A)')'ISTKRUN           LOWER ISTOKPOGA BASIN RUNOFF (AC
     $-FT)'
      WRITE(68,'(A)')'NLSDMD            NORTH LAKE SHORE BASIN SUPP DMD
     $(AC-FT)'
      WRITE(68,'(A)')'TCNSTNLS          NORTH LAKE SHORE BASIN DMD MET B
     $Y TCNSQ (AC-FT)'
      WRITE(68,'(A)')'NLSDMDMET         NORTH LAKE SHORE BASIN SUPP DMD
     $MET BY LOK (AC-FT)'
      WRITE(68,'(A)')'NLSRUN            NORTH LAKE SHORE BASIN RUNOFF (A
     $C-FT)'
      WRITE(68,'(A)')'NLSTTCNS          NORTH LAKE SHORE BASIN RUNOFF CO
     $NTRIBUTING TO TCNSQ (AC-FT)'
      WRITE(68,'(A)')'NLSTLOK           NORTH LAKE SHORE BASIN RUNOFF DI
     $RECTLY TO LOK (AC-FT)'
      WRITE(68,'(A)')'NELSDMD           NORTHEAST LAKE SHORE BASIN SUPP
     $DMD (AC-FT)'
      WRITE(68,'(A)')'NELSDMDMET        NORTHEAST LAKE SHORE BASIN SUPP
     $DMD MET BY LOK (AC-FT)'
      WRITE(68,'(A)')'NELSRUN           NORTHEAST LAKE SHORE BASIN RUNOF
     $F (AC-FT)'
      WRITE(68,'(A)')'DATA'

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  WRITE header to output file of unit no 67

      WRITE(67,'(A)')'YEAR '
      WRITE(67,'(A)')'MONTH'
      WRITE(67,'(A)')'DAY '
      WRITE(67,'(A)')'SBCYTDMD          SEMINOLE BIG CYPRESS TOT DMD (AC
     $-FT)'
      WRITE(67,'(A)')'SBCYLOCAL         SEMINOLE BIG CYPRESS DMD MET BY
     $LOCAL (S-190 FLOWS) CONTRIBUTION (AC-FT)'
      WRITE(67,'(A)')'SBCYCSDMD         SEMINOLE BIG CYPRESS SUPP DMD (A
     $C-FT)'
      WRITE(67,'(A)')'SBCYROTT          SEMINOLE BIG CYPRESS SUPP DMD ME
     $T BY ROTT WMA (AC-FT)'
      WRITE(67,'(A)')'SBCYSTA6          SEMINOLE BIG CYPRESS SUPP DMD ME
     $T BY STA6 (AC-FT)'
      WRITE(67,'(A)')'SBCYLOK           SEMINOLE BIG CYPRESS SUPP DMD ME
     $T BY LOK (AC-FT)'
      WRITE(67,'(A)')'SBRDMD            SEMINOLE BRIGHTON SUPP DMD (AC-F
     $T)'
      WRITE(67,'(A)')'SBRDMDMET         SEMINOLE BRIGHTON SUPP DMD MET B
     $Y LOK (AC-FT)'
      WRITE(67,'(A)')'SHLYWDPDMD        SEMINOLE HOLLYWOOD PWS PUMPAGE D
     $MD (MGD)'
      WRITE(67,'(A)')'SHLYWDPUMP        SEMINOLE HOLLYWOOD PWS ACTUAL PU
     $MPAGE (MGD)'
      WRITE(67,'(A)')'DATA'

c
c
c-----------------------------
```

```
c     WRITE header to srs flow.dat file
c-----------------------------
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (type_flow_across_ttrail .NE. 'MINDEL ') THEN
         IF (type_flow_across_ttrail .NE. 'STAGETG') THEN
            WRITE(151,'(A)')'YEAR '
            WRITE(151,'(A)')'MONTH'
            WRITE(151,'(A)')'DAY '
            WRITE(151,'(A)')'S12FLWTARG     S12 FLOW TARGET'
            WRITE(151,'(A)')'S333TRGUNM     FLOW TARGET UNMET BY S333'
            WRITE(151,'(A)')'S12REG         S12 REG FLOW'
            WRITE(151,'(A)')'S12NONREG      S12 NON REG FLOW'
            WRITE(151,'(A)')'S333FLTRG      S333 FLOW TARGET'
            WRITE(151,'(A)')'S333REG        S333 REG FLOW'
            WRITE(151,'(A)')'S333NONREG     S333 NON REG FLOW'
            WRITE(151,'(A)')'S355FLTRG      S355 FLOW TARGET'
            WRITE(151,'(A)')'S355REG        S355 REG FLOW'
            WRITE(151,'(A)')'S355NONREG     S355 NON REG FLOW'
            WRITE(151,'(A)')'WCA3ARGZON     WCA3A REG ZONE'
            WRITE(151,'(A)')'G3273CONST     G3273 CONSTRNT INDX(1-exceed
     $ed,0-not exceeded'
            WRITE(151,'(A)')'G3273CELL      G3273 CELL STAGE (R17C24)'
            WRITE(151,'(A)')'DATA'
         ENDIF
      ENDIF
c
c North Storage reservoir budget components
c
      IF (inorth_stor_opt .EQ. 1) THEN
         WRITE(125,'(A)')'YEAR '
         WRITE(125,'(A)')'MONTH'
         WRITE(125,'(A)')'DAY '
         WRITE(125,'(A)')'RAINFALL          RAINFALL VOLUME(ACRE-FT)'
         WRITE(125,'(A)')'SPILLOVER         EXCESS WATER ABOVE MAX DPTH'
         WRITE(125,'(A)')'POTET             MAXIMUM(POTENTIAL) ET'
         WRITE(125,'(A)')'ACTUALET          ACTUAL ET LOSS'
         WRITE(125,'(A)')'POTSEEP           POTENTIAL SEEPAGE LOSS'
         WRITE(125,'(A)')'SEEPLOSS          ACTUAL SEEPAGE LOSS'
         WRITE(125,'(A)')'TOTALLOSS         TOTAL LOSS(ET+SEEP)'
         WRITE(125,'(A)')'RESTLOK           RESERVOIR OUTFLOW TO LOK'
         WRITE(125,'(A)')'LOKTRES           LOK OUTFLOW TO RESERVOIR'
         WRITE(125,'(A)')'HRES0             DEPTH OF WATER IN RESERVOIR
     $AT BEG OF DAY'
         WRITE(125,'(A)')'HRES1             DEPTH OF WATER(FT) AFTER RAI
     $N,ET AND SEEPAGE,BUT BEFORE SPILLOVER'
         WRITE(125,'(A)')'HRES2             DEPTH OF WATER(FT) AFTER SPI
     $LLOVER'
         WRITE(125,'(A)')'HRES              DEPTH OF WATER IN RESERVOIR
     $AT END OF DAY'
         WRITE(125,'(A)')'DATA'
      ENDIF
c
c Taylor Creek/Nubbin Slough reservoir budget components
c
      IF (itcns_res_opt .EQ. 1) THEN
         WRITE(126,'(A)')'YEAR '
```

```
         WRITE(126,'(A)')'MONTH'
         WRITE(126,'(A)')'DAY '
         WRITE(126,'(A)')'RAINFALL          RAINFALL VOLUME(ACRE-FT)'
         WRITE(126,'(A)')'SPILLOVER         EXCESS WATER ABOVE MAX DPTH'
         WRITE(126,'(A)')'POTET             MAXIMUM(POTENTIAL) ET'
         WRITE(126,'(A)')'ACTUALET          ACTUAL ET LOSS'
         WRITE(126,'(A)')'POTSEEP           POTENTIAL SEEPAGE LOSS'
         WRITE(126,'(A)')'SEEPLOSS          ACTUAL SEEPAGE LOSS'
         WRITE(126,'(A)')'TOTALLOSS         TOTAL LOSS(ET+SEEP)'
         WRITE(126,'(A)')'RESTLOK           RESERVOIR OUTFLOW TO LOK'
         WRITE(126,'(A)')'LOKTRES           LOK OUTFLOW TO RESERVOIR'
         WRITE(126,'(A)')'HRES0             DEPTH OF WATER IN RESERVOIR
     $AT BEG OF DAY'
         WRITE(126,'(A)')'HRES1             DEPTH OF WATER(FT) AFTER RAI
     $N,ET AND SEEPAGE,BUT BEFORE SPILLOVER'
         WRITE(126,'(A)')'HRES2             DEPTH OF WATER(FT) AFTER SPI
     $LLOVER'
         WRITE(126,'(A)')'HRES              DEPTH OF WATER IN RESERVOIR
     $AT END OF DAY'
         WRITE(126,'(A)')'DATA'
      ENDIF
c
c  output for flow over passive weirs
c
      IF (LP(12)) THEN
         WRITE(130,'(A)')'YEAR '
         WRITE(130,'(A)')'MONTH'
         WRITE(130,'(A)')'DAY '
         WRITE(130,'(A)')'TSTEPNO           TIME STEP NO WITHIN DAY'
         WRITE(130,'(A)')'HEADW             HEADWATER FOR PASSIVE WEIR'
         WRITE(130,'(A)')'TAILW             TAILWATER FOR PASSIVE WEIR'
         WRITE(130,'(A)')'CRSTELEV          ELEVATION(FT NGVD) OF WEIR C
     $REST'
         WRITE(130,'(A)')'AVAILDPH          AVAILABLE DEPTH OF WATER FOR
     $ OUTFLOW OVER PASSIVE WEIR'
         WRITE(130,'(A)')'STRUCCAP          WEIR CAPACITY(CFS-TIMESTEP)'
         WRITE(130,'(A)')'AVAILVOL          AVAILABLE VOLUME (CFS-TIMEST
     $EP) OF WATER FOR OUTFLOW'
         WRITE(130,'(A)')'WIERFLOW          WIER OUTFLOW(CFS-TIMESTEP)'
         WRITE(130,'(A)')'DSTGCNSTRT        STAGE DOWNSTREAM OF WIER AT
     $TRIGGER LOCATION FOR CONSTRAINING FLOW'
         WRITE(130,'(A)')'MAXDSTGWR         MAXIMIM DOWNSTREAM STAGE AT
     $TRIGGER LOCATION FOR FLOW OVER WIER'
         WRITE(130,'(A)')'WIERNAME          NAME (MAX 6 CHARACTERS) OF
     $PASSIVE WIER'
         WRITE(130,'(A)')'DATA'
      ENDIF
C----------------------------------------
C     WRITE header to daily_asr_budg.dat
C----------------------------------------
      IF (num_asr_output .gt. 0) THEN
         WRITE(65,'(A)')'YEAR '
         WRITE(65,'(A)')'MONTH'
         WRITE(65,'(A)')'DAY  '
         DO iasr = 1,num_asr_output
           IF (nres_src_to_asr(iasr) .gt. 0) THEN
             char_id_asr = charid_asr(iasr,1)
```

```
            DO ll = 1,LEN(char_id_asr)
              IF (char_id_asr(ll:ll).eq. ' ') THEN
                 char_id_asr(ll:ll) = '_'
              ENDIF
            ENDDO
            WRITE(65,665) char_id_asr
 665        FORMAT(A6,'_VOL        ASR BUBBLE VOLUME(1000 ACRE-FT)')
            WRITE(65,666) char_id_asr
 666        FORMAT(A6,'_INJ        VOLUME OF WATER INJECTED INTO ASR(IN
     $ 1000 ACRE-FT)')
            WRITE(65,667) char_id_asr
 667        FORMAT(A6,'_LOS        VOLUME OF WATER LOST IN THE INJECTIO
     $N DUE TO INEFFICIENCY(IN 1000 ACRE-FT)')
            WRITE(65,668) char_id_asr
 668        FORMAT(A6,'_REC        VOLUME OF WATER RECOVERED FROM ASR W
     $ELLS FOR ENVIR/WS PURPOSES(IN 1000 ACRE-FT)')
          ELSEIF (ncnl_src_to_asr(iasr) .gt. 0) THEN
            char_id_cnl = cnm(cnl_src_num(iasr,1))
            DO ll = 1,LEN(char_id_cnl)
              if (char_id_cnl(ll:ll).eq. ' ') THEN
                 char_id_cnl(ll:ll) = '_'
              endif
            ENDDO
            WRITE(65,669) char_id_cnl
            WRITE(65,670) char_id_cnl
            WRITE(65,671) char_id_cnl
            WRITE(65,672) char_id_cnl
 669        FORMAT(A5,'A_VOL        ASR BUBBLE VOLUME(1000 ACRE-FT)')
 670        FORMAT(A5,'A_INJ        VOLUME OF WATER INJECTED INTO ASR(I
     $N 1000 ACRE-FT)')
 671        FORMAT(A5,'A_LOS        VOLUME OF WATER LOST IN THE INJECTI
     $ON DUE TO INEFFICIENCY(IN 1000 ACRE-FT)')
 672        FORMAT(A5,'A_REC        VOLUME OF WATER RECOVERED FROM ASR
     $WELLS FOR ENVIR/WS PURPOSES(IN 1000 ACRE-FT)')
          ELSE
            WRITE(65,'(A)')'LOKASR_VOL        ASR BUBBLE VOLUME(1000 AC
     $RE-FT)'
            WRITE(65,'(A)')'LOKASR_INJ        VOLUME OF WATER INJECTED
     $INTO ASR(IN 1000 ACRE-FT)'
            WRITE(65,'(A)')'LOKASR_LOS        VOLUME OF WATER LOST IN T
     $HE INJECTION DUE TO INEFFICIENCY(IN 1000 ACRE-FT)'
            WRITE(65,'(A)')'LOKASR_REC        VOLUME OF WATER RECOVERED
     $ FROM ASR WELLS FOR ENVIR/WS PURPOSES(IN 1000 ACRE-FT)'
          ENDIF
        ENDDO
        WRITE(65,'(A)')'DATA'
      ENDIF
C----------------------------------------
C     WRITE header to daily_lok_et.dat file
C
C----------------------------------------
      IF (LP(10)) THEN
       IF (runmode .eq. 'SIMUL') then
        WRITE(181,'(A)')'YEAR '
        WRITE(181,'(A)')'MONTH'
        WRITE(181,'(A)')'DAY  '
        WRITE(181,'(A)')'LOKSTG         LAKE OKEECHOBEE STAGE(FT NGVD)'
```

```
         WRITE(181,'(A)')'OPNETVOL       VOLUME OF EVAPORATION FROM OPEN
     $ WATER PORTION OF LOK(ACRE_FT)'
         WRITE(181,'(A)')'LITETVOL       VOLUME OF ET FROM WET PORTION O
     $F LITTORAL ZONE IN LOK'
         WRITE(181,'(A)')'DRYETVOL       VOLUME OF ET FROM DRY PORTION O
     $F LITTORAL ZONE IN LOK'
         WRITE(181,'(A)')'TOTETVOL       TOTAL VOLUME(ACRE-FT) OF ET FRO
     $M LOK (OPNETVOL+LITETVOL+DRYETVOL)'
         WRITE(181,'(A)')'DATA'
       ELSE
         WRITE(181,'(A)')'YEAR '
         WRITE(181,'(A)')'MONTH'
         WRITE(181,'(A)')'DAY  '
         WRITE(181,'(A)')'LOKSTG         LAKE OKEECHOBEE STAGE(FT NGVD)'
         WRITE(181,'(A)')'TOTETVOL       TOTAL VOLUME(CFS-DAY) OF ET FRO
     $M LOK (OPNETVOL+LITETVOL+DRYETVOL)'
       ENDIF
      ENDIF
C-----------------
      RETURN
   10 FORMAT(14I3)
   80 FORMAT (1H1,T10,'SOUTH FLORIDA WATER MANAGEMENT MODEL', T110,
     $'RUN DATE ',A24//, ' PERIOD OF SIMULATION ',T55,I4,' -- ',I4/,
     $' LENGTH OF TIME STEP (DAYS) ',T55,I4/,
     $' NUMBER OF ROWS OF NODES IN NORTH-SOUTH DIRECTION ',T55,I4/,
     $' MAXIMUM NUMBER OF COLUMNS PER ROW',T55,I4/,
     $' TOTAL NUMBER OF GRID POINTS ',T55,I4/,
     $' EAST WEST GRID SPACING (MILES) ',T55,I4/,
     $' NORTH SOUTH GRID SPACING (MILES) ',T55,I4/,
     $' TOTAL NUMBER OF LAND USE TYPES',T55,I4/,
     $' NUMBER OF KNOWN FLOW POINTS READ IN ',T55,I4/,
     $' NUMBER OF KNOWN WELL FIELD PUMPAGES ',T55,I4/,
     $' SURFACE DETENTION DEPTHS',T55,15F5.2/,
     $' PRINT OPTIONS ( 1 = YES) ',T55,15I3)
c
   90 FORMAT (/T15,'MANNINGS COEFFICIENTS',T56,'SRZ',T65,'DRZ',/,
     $' URBAN',T30,F8.4,T40,F8.4,T54,F5.2,T63,F5.2/, ' AGRICULTURAL',
     $T30,F8.4,T40,F8.4,T54,F5.2,T63,F5.2/, ' SWAMP I',T30,F8.4,T40,F8.
     $4,T54,F5.2,T63,F5.2/, ' VACANT LAND',T30,F8.4,T40,F8.4,T54,F5.2,
     $T63,F5.2/, ' BIG CYPRESS',T30,F8.4,T40,F8.4,T54,F5.2,T63,F5.2/,
     $' SWAMP II',T30,F8.4,T40,F8.4,T54,F5.2,T63,F5.2/,
     $' AGRICULTURAL--TRUCK CROPS',T30,F8.4,T40,F8.4,T54,F5.2, T63,F5.2
     $/, ' AGRICULTURAL--SUGARCANE  ',T30,F8.4,T40,F8.4,T54,F5.2, T63,F5
     $.2/, ' AG -- IRRIGATED PASTURE',T30,F8.4,T40,F8.4,T54,F5.2, T63,F5
     $.2)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
  100 FORMAT (/T40,'MINX ARRAY',/5(5X,20I4,/))
  110 FORMAT (T40,'MAXX ARRAY',/5(5X,20I4,/))
  120 FORMAT (T40,'MXOV ARRAY',/5(5X,20I4,/))
  130 FORMAT(1x,(A6,1x),'       MARSH STAGE IN WCA1 AT OR BELOW FLOOR(1-
     $YES,0-NO)')
  131 FORMAT(1x,(A6,1x),'       MARSH STAGE IN WCA2A AT OR BELOW FLOOR(1
     $-YES,0-NO)')
  132 FORMAT(1x,(A6,1x),'       MARSH STAGE IN WCA3A AT OR BELOW FLOOR(1
     $-YES,0-NO)')
      END
```

open_binary_output_files.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: open_binary_output_files.F,v 1.18 2005/06/10 13:29:15 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/open_binary_output_files.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE OPENS NECESSARY BINARY OUTPUT FILES AND
C   WRITES HEADERS TO FILES. THE FILES ARE SET UP SO THAT TIME
C   DEPENDENT DATA IN BINARY FORMAT CAN BE OUTPUT TO FILES.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE open_binary_output_files (runtitle, getfdd, ip,
     & isum_nsm, minx_nsm, maxx_nsm, maxy_nsm, isum_w_lok, minx_w_lok,
     & maxx_w_lok, maxylok, numd,use_lec_et, ibm, ism, isyr, ifltabs,
     & ifltabc, rain_def, title_rain)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'

      CHARACTER dssfn*40, filename*200, runtitle*256, title_rain*70

      INTEGER getfdd(150), ifltabc(1200), ifltabs(1200), ip(60),
     & isum_nsm(max_n_rows_incl_lok), isum_w_lok(max_n_rows_incl_lok),
     & maxx_nsm(max_n_rows_incl_lok), maxx_w_lok(max_n_rows_incl_lok),
     & minx_nsm(max_n_rows_incl_lok), minx_w_lok(max_n_rows_incl_lok),
     & numd(12), rain_def, use_lec_et

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      maxylok=71

      CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX, MAXX,
```

```
$ISUM)

 IF (IP(20).eq.1 .AND. IP(22).eq.1) THEN
 IF (IP(19).eq.1) THEN
   FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$  '/daily_surface_flw.bin'
     ELSE
   FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$  '/mthly_tot_surface_flw.bin'
   ENDIF
   CALL openfilef77(43,FILENAME,"wb")
   IF (IP(25).eq.1) THEN
    FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$   '/daily_gw_flw.bin'
   ELSE
    FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$   '/mthly_tot_gw_flw.bin'
   ENDIF
   CALL openfilef77(49,FILENAME,"wb")
 ENDIF
 IF (IP(16).eq.1 .AND. IP(22).eq.1) THEN
   FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/eomth_stage.bin'
   CALL openfilef77(40,FILENAME,"wb")
   FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$  '/eomth_avg_cell_stage.bin'
   CALL openfilef77(140,FILENAME,"wb")
   FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/eomth_cell_stor.bin'
   CALL openfilef77(142,FILENAME,"wb")
   FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/mthly_asrrec_vol.bin'
   CALL openfilef77(143,FILENAME,"wb")
 ENDIF
 IF (IP(17).eq.1 .AND. IP(22).eq.1) THEN
   FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/eomth_ponding.bin'
   CALL openfilef77(41,FILENAME,"wb")
 ENDIF
 IF (IP(18).eq.1 .AND. IP(22).eq.1) THEN
   FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/mthly_tot_et.bin'
   CALL openfilef77(42,FILENAME,"wb")

   FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$  '/mthly_tot_rainfall.bin'
   CALL openfilef77(44,FILENAME,"wb")
   FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$  '/mthly_tot_pumpage.bin'
   CALL openfilef77(45,FILENAME,"wb")
   IF (IP(25).eq.1) THEN
    FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/daily_unsatdph.bin'
   ELSE
    FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/eomth_unsatdph.bin'
   ENDIF
   CALL openfilef77(50,FILENAME,"wb")
 ENDIF
 IF( ip(22).eq.1 .AND. ip(23).eq.1) THEN
    FILENAME = DATADIR(1:LENGTH(DATADIR)) //
$   '/daily_stg_minus_lsel.bin'
   CALL openfilef77(47,FILENAME,"wb")
 ENDIF
```

```
c
c  daily ET file for all grid cells,units in inches
c
      IF (ip(22).eq.1 .AND. ip(21).eq.1) THEN
          FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $   '/daily_tot_et.bin'
         CALL openfilef77(127,FILENAME,"wb")
      ENDIF
c
      IF (IP(22).eq.1 .AND. IP(24).eq.1) THEN
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $  '/mthly_tot_ovflw_to_canal.bin'
        CALL openfilef77(39,FILENAME,"wb")
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $  '/mthly_tot_seep_to_canal.bin'
        CALL openfilef77(35,FILENAME,"wb")
         FILENAME = DATADIR(1:LENGTH(DATADIR)) //
     $  '/mthly_tot_canal_evap.bin'
        CALL openfilef77(38,FILENAME,"wb")
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (IP(20).eq.1 .AND. IP(22).eq.1) THEN
        getfdd(43) = 43
        getfdd(49) = 49
      ENDIF
      IF (IP(16).eq.1 .AND. IP(22).eq.1) THEN
        getfdd(40) = 40
        getfdd(140) = 140
        getfdd(142) = 142
        getfdd(143) = 143
      ENDIF
      IF (IP(17).eq.1 .AND. IP(22).eq.1) THEN
        getfdd(41) = 41
      ENDIF
      IF (IP(18).eq.1 .AND. IP(22).eq.1) THEN
        getfdd(42) = 42
        getfdd(44) = 44
        getfdd(45) = 45
        getfdd(50) = 50
      ENDIF
      IF( ip(22).eq.1 .AND. ip(23).eq.1) THEN
         getfdd(47) = 47
      ENDIF
      IF (ip(22).eq.1 .AND. ip(21).eq.1) THEN
         getfdd(127) = 127
      ENDIF
      IF (ip(22).eq.1 .and. ip(24).eq.1) THEN
         getfdd(39) = 39
         getfdd(35) = 35
         getfdd(38) = 38
      ENDIF
c
      IF (IP(20).eq.1 .AND. IP(22).eq.1) THEN
        CALL GRIDWHD(43, IERR)
        CALL GRIDWHD(49, IERR)
      ENDIF
      IF (IP(16).eq.1 .AND. IP(22).eq.1) THEN
```

```
        CALL GRIDWHD(40, IERR)
        CALL GRIDWHD(140, IERR)
        CALL GRIDWHD(142, IERR)
        CALL GRIDWHD(143, IERR)
      ENDIF
      IF (IP(17).eq.1 .AND. IP(22).eq.1) THEN
        CALL GRIDWHD(41, IERR)
      ENDIF
      IF (IP(18).eq.1 .AND. IP(22).eq.1) THEN
        CALL GRIDWHD(42, IERR)
        CALL GRIDWHD(44, IERR)
        CALL GRIDWHD(45, IERR)
        CALL GRIDWHD(50, IERR)
      ENDIF
      IF( ip(22).eq.1 .AND. ip(23).eq.1) THEN
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         call val2val_w_lok(0,0.0,0.0,isum_w_lok,minx_w_lok,maxx_w_lok)
         CALL SETGRID(RUNTITLE,maxylok,isum_w_lok(maxylok+1),DX,DY,
     $   minx_w_lok,maxx_w_lok,isum_w_lok)
c
         CALL GRIDWHD(47, IERR)
      ENDIF
      IF (ip(22).eq.1 .AND. ip(21).eq.1) THEN
         CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX, MAXX,
     $   ISUM)
         CALL GRIDWHD(127, IERR)
      ENDIF
      IF (IP(22).EQ.1 .AND. IP(24).EQ.1) THEN
         CALL SETGRID(RUNTITLE, MAXY, ISUM(MAXY+1), DX, DY, MINX, MAXX,
     $   ISUM)
         CALL GRIDWHD(39, IERR)
         CALL GRIDWHD(35, IERR)
         CALL GRIDWHD(38, IERR)
      ENDIF

      IF (use_lec_et .NE. 0 ) THEN
         FILENAME= DATADIR(1:LENGTH(DATADIR)) //
     $  '/mthly_tot_et_unsat_unacct.bin'

         CALL openfilef77(57,FILENAME,"wb")
         getfdd(57) = 57
         call setgrid('SFWMM Unaccounted for Unsaturated ET Data ',
     $   maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
         CALL GRIDWHD(57, IERR)
      ENDIF
c
      IF (use_lec_et .NE. 0 .and. ip(18).eq. 1 .and. ip(22).eq.1) THEN
       IF ( ip(25).eq.1 ) THEN
         FILENAME=DATADIR(1:LENGTH(DATADIR)) //
     $  '/daily_infilt_perc.bin'
       ELSE
         FILENAME=DATADIR(1:LENGTH(DATADIR)) //
     $  '/mthly_tot_infilt_perc.bin'
       ENDIF
         CALL openfilef77(46,FILENAME,"wb")
         getfdd(46) = 46
```

```
         call setgrid('SFWMM Infiltration and Percolation Data   ',
     $   maxy, isum(maxy+1), dx, dy, minx, maxx, isum)
         CALL GRIDWHD(46, IERR)
      ENDIF
c
C   INPUT HEADER FOR RAINFALL FILE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      getfdd(7) = 7
      rain_def = 7
      call gridrhd(rain_def, ierr)
      call getgrid(title_rain,maxy_nsm,isum_nsm(maxy_nsm + 1),dx,dy ,
     $minx_nsm,maxx_nsm,isum_nsm)
C-----------------
C   POSITION  RF FILE TO CORRECT STARTING DATE
C-----------------
c have to be done with et_input and nirrdmd binaries also !!!
c BUT NOT HERE! That is done in the subroutine lec_et_initial_read
c
c Only need to call setgrid before 1st gread to a specific file
c
      call setgrid(title_rain,maxy_nsm,isum_nsm(maxy_nsm+1),dx,dy ,
     $minx_nsm, maxx_nsm,isum_nsm)
      nskip0 = iymdjl(ifyr,ibm,1) - iymdjl(isyr,ism,1)
      NMTHS = (IFYR - ISYR) * 12 + (IBM - ISM)
      IF (NMTHS .GT. 0) THEN
         DO 10 I = ISM,ISM+NMTHS-1
            NY = (I - 1) / 12 + 1
            NUMD(2) = 28
            IF(MOD(ISYR+NY-1,4) .EQ. 0) NUMD(2) = 29
            MTH = I - (NY - 1) * 12
   10    CONTINUE
      ENDIF
c
c skip daily rainfall
      IF (nskip0.GT.0)THEN
         call gridskp(rain_def, nskip0, ierr)
      ENDIF
c-----------------------------------
C OPEN DSSFILE
C-----------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      CALL ZSET('MLEVEL','',4)
c
c  open str2x2.dss
      DSSFN='daily_str_flw.dss'
      FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/' // DSSFN
      CALL ZOPEN(IFLTABS,FILENAME,ISTAT)
      IF (ISTAT.NE.0) THEN
         WRITE (*,*) 'Something is wrong with ',DSSFN,'!!'
      ENDIF
c
c  open canal.dss
c
      DSSFN='daily_canal_stg.dss'
      FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/' // DSSFN
      CALL ZOPEN(IFLTABC,FILENAME,ISTAT)
```

```
IF (ISTAT.NE.0) THEN
   WRITE (*,*) 'Something is wrong with ',DSSFN,'!!'
ENDIF
RETURN
END
```

open_input_data_files.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: open_input_data_files.F,v 1.11 2003/07/03 20:07:39 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/open_input_data_files.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE OPENS THE INPUT DATA FILES
C   THE PATHS ARE INPUT IN ALTWMM FILE
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE open_input_data_files (runtitle, infile, infil1,
     & dmdronm, ccdmdro, cfcurfut, ifltab1, ifltabr)

      CHARACTER*200 alt, altdef, arg(1), infil1, infile
      CHARACTER ccdmdro(2)*6, cfcurfut(2)*7, dmdronm(10)*10,
     & runtitle*256

      INTEGER ifltab1(1200), ifltabr(1200), infl(50)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   OPEN ALL DATA FILES THROUGH INPUT

      CALL ZSET('MLEVEL','',4)

      nargs = iargc()
      IF (nargs .EQ. 0) THEN
         altdef = 'ALTWMM'
      ELSE
         call getarg(1,arg(1))
         altdef = arg(1)
      ENDIF
```

```
c      OPEN (51,file = altdef)
c     call subroutine that open cleaned file
      call remov_comm_inp_files(altdef,51)
      READ (51,'(A)') RUNTITLE
      READ (51,*)     NFILES
      DO I = 1,NFILES
         READ (51,'(I3,2x,A)') INFL(I),ALT
         L = LENGTH(ALT)
         INFILE = ALT(1:L)
         if(i .EQ. 3) THEN
            CALL openfilef77(infl(3),INFILE,"rb")
         ELSE
c           OPEN(INFL(I), FILE = INFILE)
c           call subroutine that open and clean file if is local
            call remov_comm_inp_files(INFILE,INFL(I))
         ENDIF
      ENDDO
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C one final last read: location of flo.dss file
c
      READ (51,'(A)') ALT
      L = LENGTH(ALT)
      INFILE = ALT(1:L)
c
c one ultimately last read: location of dmdro2x2.dss file
c
      READ (51,'(A)') ALT
      L = LENGTH(ALT)
      INFIL1 = ALT(1:L)
c
      CLOSE(51)
      PRINT *,INFILE,INFIL1
c
c  OPEN flo.dss and dmndro2x2.dss
c
      call zOPEN(ifltabr,infile,istat)
      IF (istat .NE. 0) THEN
         WRITE (*,*) 'Something is wrong with ',infile,'!!'
      ENDIF
      call zOPEN(ifltab1,infil1,istat)
      IF (istat.NE.0) THEN
         WRITE (*,*) 'Something is wrong with ',infile,'!!'
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
cc define B part of DSS pathname for the dmnd/ro file
c
      dmdronm(1)='STLUCIE'
      dmdronm(2)='CALOOS'
      dmdronm(3)='ISTOKPBB'
      dmdronm(4)='S4DISTON'
      dmdronm(5)='S4OTHER'
      dmdronm(6)='NLKSHORE'
      dmdronm(7)='NELKSHORE'
      dmdronm(8)='ISTOKPAB'
cc define C part of DSS pathname for the dmnd/ro file
```

```
      ccdmdro(1) = 'RUNOFF'
      ccdmdro(2) = 'DEMAND'
cc define F part of DSS pathname for the dmnd/ro file
      cfcurfut(1) = 'CURRENT'
      cfcurfut(2) = 'FUTURE'
      RETURN
      END
```

oper_sched_data.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: oper_sched_data.F,v 1.14 2004/01/20 15:52:28 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/oper_sched_data.F,v $
C   $Name: rel-5-5 $
C
C
C   SUBROUTINE TO RETRIEVE DATA DEFINING OPERATIONAL SCHEDULES FOR LAKE
C       OKEECHOBEE,WCAs,HOLEYLAND,AND ROTENBERGER TRACT
C
C   INPUT FILES USED:
c    lok_wca_oper_sched.dat  INPUT FILE FOR OPERATIONAL SCHEDULES
C       FOR LOK,WCAS,HOLEYLAND,AND ROTENBERGER
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE oper_sched_data (clim_threshold_est, offset_to_sched,
     $ wup0, wup, numd, ssm_vol_cutback_thres, month_end_credit,
     & nbrkpt_ssm1, iopt_for_semcyp_prior_roten, ibeg_mth_lokdev,
     & iend_mth_lokdev, ibeg_day_lokdev, iend_day_lokdev, etp, rfp,
     & iopt_drawdown,regstgwt, regstgwn, nbrkpt_ssmwt, nbrkpt_ssmwn,
     & iregjulwt, iregjulwn, ibot_zone_regd)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'climvar.inc'
      INCLUDE 'loktarg.inc'
```

```
      DIMENSION clim_threshold_est(2),
     & ibot_zone_regd(max_n_stor_w_oper_sch),
     & idayreg(15),regstgwt(25),regstgwn(25), mthreg(15),
     & numd(12), offset_to_sched(max_n_stor_w_oper_sch)

      REAL etp(12), iregjulwn(25), iregjulwt(25), rfp(12), wup(12),
     & wup0(12)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C     READ IN BREAKPOINT DATA FOR OPERATIONAL SCHEDULES FOR LOK AND WCAS

      READ (102,*) n_stor_areas,no_add_areas_to_wcas

      DO i = 1,n_stor_areas
         READ(102,'(A6,2x,7i5)') stor_area_name(i),nzone(i) ,
     $   itop_zone_indx(i),ibot_zone_indx(i) ,ialt_bot_zone_indx(i),
     $   ibot_zone_indx_s333reg(i),ifloor_line_indx(i),
     $   ibot_zone_regd(i)
         IF (stor_area_name(i) .EQ. 'LOK   ') THEN
            READ (102,*) iclimate_opt,itrib_hydro_cond_opt ,
     $      multi_seas_pred_opt,izone_flex_eaares_opt ,
     $      iflex_clim_lokreg_to_calstlres,iadd_ops_to_eaa_res_line
            READ (102,'(1x,8i3,2x,a6)') n_fc_zones,no_puls_zones ,
     $      mzone_puls_top,mzone_puls_bot,izone_bot_flood_flows_south ,
     $      izone_bot_state_eaares,iflex_pulse_opt,iplsday,
     $      opt_sim_lok_eaares
            READ (102,'(I3,20F6.0)') no_puls_levels,(avg_pulse_release
     $      (k,1) ,avg_pulse_release(k,2),k=1,no_puls_levels)
            READ (102,'(8F7.0)') (s65e_inflw_thres_high(j),j=1,2) ,
     $      (s65e_inflw_thres_low(j),j=1,2) ,trib_rfet_thres,
     $      s65e_runff_wkly_thres ,(clim_threshold_est(j),j=1,2)
            READ (102,*) s_forec_eaar_thres1,s_forec_eaar_thres2

            READ (102,'(3F7.1)') cresbp_max_rate,cal_res_dpth_thres_bp ,
     $      rmax_stl_res_bflw_cap
            READ (102,*) ssm_vol_cutback_thres,month_END_credit
            READ (102,*) ibeg_mth_lokdev,ibeg_day_lokdev,iend_mth_lokdev
     $      ,iend_day_lokdev,iopt_drawdown
            READ (102,'(4F7.2)') (offset_reg_to_wcas(j),j=1,4)
            DO j = 1,nzone(i)
               IF (j .LE. mzone_puls_bot .OR. (no_puls_zones .EQ. 0
     $         .AND. j .LE. n_fc_zones)) THEN
                  READ (102,'(A7)') zoneid(j)
                  READ(102,*) (rmax_out_capac_wet(j,ii) ,
     $            rmax_out_capac_norm(j,ii),ii=1,2)
                  IF (j .GE. mzone_puls_top .AND. j .LE. mzone_puls_bot)
     $            THEN
                     IF (j .EQ. mzone_puls_top) THEN
                        IF (mzone_puls_bot - mzone_puls_top + 1 .LT.
     $                  no_puls_levels) THEN
                           DO kk = no_puls_levels,mzone_puls_bot
     $                     -mzone_puls_top + 2,-1
                              iplslevel = kk
                              READ (102,*) no_of_pulse_rel(iplslevel) ,
     $                        (qplsl(iplslevel,k),k=1,no_of_pulse_rel
```

```
     $                        (iplslevel))
                              READ(102,*) (qplscl(iplslevel,k) ,k=1,
     $                        no_of_pulse_rel(iplslevel))
                           END DO
                        ENDIF
                     ENDIF
                     iplslevel = mzone_puls_bot - j + 1
                     READ(102,*) no_of_pulse_rel(iplslevel),(qplsl
     $               (iplslevel,k) ,k=1,no_of_pulse_rel(iplslevel))
                     READ(102,*) (qplscl(iplslevel,k),k=1,
     $               no_of_pulse_rel (iplslevel))
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  READ(102,*) nbrkpt(i,j),(mthreg(k),idayreg(k) ,k = 1,
     $            nbrkpt(i,j))
                  DO k = 1,nbrkpt(i,j)
                     iregjul(i,j,k) = julian(mthreg(k),idayreg(k),numd)
                  END DO
                  READ(102,*) (regstg(i,j,k),k=1,nbrkpt(i,j))
                  READ(102,'(4(A4,2x),4F6.2,2I5,2x,A7,2i5)')
     $            (eaa_conv_opt_reg(k,j,1),k=1,2) ,(eaa_conv_opt_reg
     $            (k,j,2),k=1,2),(rmin_clim_indx_thres
     $            (j,k,1),k=1,2) ,(frac_depth_zone(j,k),k=1,2),
     $            ipulse_level_in_zone(j) ,iopt_for_interp(j),
     $            opt_for_pulsing(j) ,(igrav_sim_opt(k,j),k=1,2)

                  READ(102,'(4F7.2)') (trib_rf_et_thres(k,j),k=1,3)
                  READ (102,'(10f7.0)') (s65e_runff_thres(k,j),k=1,3)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_trib_hydro_categ(k,j),k=1,no_of_categ)
                  READ (102,*) (ipulse_level_trib_hyd(k,j,1),k=1,
     $            no_of_categ)
                  READ (102,*) (ipulse_level_trib_hyd(k,j,2),k=1,
     $            no_of_categ)
                  READ (102,'(10f7.0)') (rmin_clim_indx_thres(j,k,2)
     $            ,k=1,2)
                  READ (102,'(10f7.0)') (rmulti_seas_thres(k,j,1),k=1,4)
                  READ (102,'(10f7.0)') (rmulti_seas_thres(k,j,2),k=1,4)
                  READ (102,*) (ipulse_level_multi_seas(k,j,1),k=1,4)
                  READ (102,*) (ipulse_level_multi_seas(k,j,2),k=1,4)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ(k,j,1),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ(k,j,2),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ(k,j,1),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ(k,j,2),k=1,no_of_categ)
                  READ (102,'(4F7.1)') (trib_rf_et_thres_wca(k,j),k=1,3)
                  READ (102,'(10f7.0)') (s65e_runff_thres_wca(k,j),k=1,
     $            3)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_trib_hydro_categ_wca(k,j),k=1,no_of_categ)
                  READ (102,'(4F7.1)') (rmin_clim_indx_thres_wca(j,k),k
     $            =1,2)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ_wca(k,j),k=1,no_of_categ)
```

```
                  READ (102,'(10f7.1)') (rmulti_seas_thres_wca(k,j),k=1,
     $            4)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ_wca(k,j),k=1,no_of_categ)
                  READ(102,'(4(A6,2x))') (limit_reg_rel_glades(k,j,1) ,k
     $            =1,4)
                  READ(102,'(4(A6,2x))') (limit_reg_rel_glades(k,j,2) ,k
     $            =1,4)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               ELSE IF (j .GT. mzone_puls_bot .AND. j .LE. n_fc_zones
     $         .AND. no_puls_zones .GT. 0) THEN
                  READ(102,*) nbrkpt(i,j),(mthreg(k),idayreg(k) ,k = 1,
     $            nbrkpt(i,j))
                  DO k = 1,nbrkpt(i,j)
                     iregjul(i,j,k) = julian(mthreg(k),idayreg(k),numd)
                  END DO
                  READ (102,*) (regstg(i,j,k),k=1,nbrkpt(i,j))
                  READ(102,'(4(A4,2x),4F6.2,2I5,2x,A7,2i5)')
     $            (eaa_conv_opt_reg(k,j,1),k=1,2) ,
     $            (eaa_conv_opt_reg(k,j,2),k=1,2),(rmin_clim_indx_thres
     $            (j,k,1),k=1,2) ,(frac_depth_zone(j,k),k=1,2),
     $            ipulse_level_in_zone(j) ,iopt_for_interp(j),
     $            opt_for_pulsing(j) ,(igrav_sim_opt(k,j),k=1,2)

                  READ(102,'(4F7.1)') (trib_rf_et_thres(k,j),k=1,3)
                  READ (102,'(10f7.1)') (s65e_runff_thres(k,j),k=1,3)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_trib_hydro_categ(k,j),k=1,no_of_categ)
                  READ (102,'(10f7.0)') (rmin_clim_indx_thres(j,k,2)
     $            ,k=1,2)
                  READ (102,'(10f7.1)') (rmulti_seas_thres(k,j,1),k=1,4)
                  READ (102,'(10f7.0)') (rmulti_seas_thres(k,j,2),k=1,4)
                  READ (102,*) (ipulse_level_multi_seas(k,j,1),k=1,4)
                  READ (102,*) (ipulse_level_multi_seas(k,j,2),k=1,4)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ(k,j,1),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ(k,j,2),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ(k,j,1),k=1,no_of_categ)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ(k,j,2),k=1,no_of_categ)
                  READ (102,'(4F7.1)') (trib_rf_et_thres_wca(k,j),k=1,3)
                  READ (102,'(10f7.0)') (s65e_runff_thres_wca(k,j),k=1,
     $            3)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_trib_hydro_categ_wca(k,j),k=1,no_of_categ)
                  READ (102,'(4F7.1)') (rmin_clim_indx_thres_wca(j,k),k
     $            =1,2)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_seas_categ_wca(k,j),k=1,no_of_categ)
                  READ (102,'(10f7.1)') (rmulti_seas_thres_wca(k,j),k=1,
     $            4)
                  READ (102,'(I5,2x,5(A7,2x))') no_of_categ ,
     $            (cgen_multi_seas_categ_wca(k,j),k=1,no_of_categ)
                  READ(102,'(4(A6,2x))') (limit_reg_rel_glades(k,j,1) ,k
     $            =1,4)
```

```
                 READ(102,'(4(A6,2x))') (limit_reg_rel_glades(k,j,2) ,k
     $           =1,4)
              ELSE
                 READ(102,*) nbrkpt(i,j),(mthreg(k),idayreg(k) ,k = 1,
     $           nbrkpt(i,j))
                 DO k = 1,nbrkpt(i,j)
                    iregjul(i,j,k) = julian(mthreg(k),idayreg(k),numd)
                 END DO
                 READ (102,*) (regstg(i,j,k),k=1,nbrkpt(i,j))
              ENDIF
           END DO
           READ(102,*) nbrkpt_ssmwt,(mthreg(k),idayreg(k) ,k = 1,
     $     nbrkpt_ssmwt)
           READ (102,*) (regstgwt(k),k=1,nbrkpt_ssmwt)
           DO k = 1,nbrkpt_ssmwt
              iregjulwt(k) = julian(mthreg(k),idayreg(k),numd)
           ENDDO
           READ(102,*) nbrkpt_ssmwn,(mthreg(k),idayreg(k) ,k = 1,
     $     nbrkpt_ssmwn)
           READ (102,*) (regstgwn(k),k=1,nbrkpt_ssmwn)
           DO k = 1,nbrkpt_ssmwn
              iregjulwn(k) = julian(mthreg(k),idayreg(k),numd)
           ENDDO
           READ(102,*) nbrkpt_ssm,(mthreg(k),idayreg(k) ,k = 1,
     $     nbrkpt_ssm)
           DO k = 1,nbrkpt_ssm
              iregjul1(k) = julian(mthreg(k),idayreg(k),numd)
           END DO
           READ (102,*) (regstg1(k),k=1,nbrkpt_ssm)
           READ (102,*) nbrkpt_ssm1,(mthreg(k),idayreg(k) ,k = 1,
     $     nbrkpt_ssm1)
           DO k = 1,nbrkpt_ssm1
              iregjul2(k) = julian(mthreg(k),idayreg(k),numd)
           END DO
           READ (102,*) (regstg2(k),k=1,nbrkpt_ssm1)
        ELSE
           DO j = 1,nzone(i)
              READ(102,*) nbrkpt(i,j),(mthreg(k),idayreg(k) ,k = 1,
     $        nbrkpt(i,j))
              DO k = 1,nbrkpt(i,j)
                 iregjul(i,j,k) = julian(mthreg(k),idayreg(k),numd)
              END DO
              READ (102,*) (regstg(i,j,k),k=1,nbrkpt(i,j))
           END DO
        ENDIF
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c   READ in breakpoints for Holeyland and Rotenberger schedules
c
      DO i = 1,no_add_areas_to_wcas
        istor_index = n_stor_areas + i
        READ(102,'(A6,2x,i5,f6.2,i5)') stor_area_name(istor_index) ,
     $  nzone(istor_index),offset_to_sched(istor_index) ,
     $  iopt_for_semcyp_prior_roten
        DO j = 1,nzone(istor_index)
           READ(102,*) nbrkpt(istor_index,j),(mthreg(k),idayreg(k) ,k =
```

```
     $      1,nbrkpt(istor_index,j))
            DO k = 1,nbrkpt(istor_index,j)
               iregjul(istor_index,j,k) = julian(mthreg(k),idayreg(k) ,
     $         numd)
            END DO
            READ (102,*) (regstg(istor_index,j,k),k= 1,nbrkpt
     $      (istor_index,j))
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c   READ estimated water use for LOK service area for determination
c    of side management cutbacks
c
      DO im = 1,12
         READ (102,*) wup0(im),wup(im),rfp(im),etp(im)
      END DO
c
c   READ in projected ET for LOK for 1,3,6 months duration
c
      IF (iclimate_opt .EQ. 1) THEN
         DO im = 1,12
            READ (102,*) totloketvol1(im),totloketvol3(im) ,totloketvol6
     $      (im),avg_demand_lok3(im)
         END DO
      ENDIF
      RETURN
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
opersched.inc

c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: opersched.inc,v 1.9 2004/01/20 15:52:28 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/opersched.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /opersched/ nzone(max_n_stor_w_oper_sch),
     &nbrkpt(max_n_stor_w_oper_sch,max_n_oper_lines),
     &iregjul(max_n_stor_w_oper_sch,max_n_oper_lines,max_n_bkpts),
     &regstg(max_n_stor_w_oper_sch,max_n_oper_lines,max_n_bkpts),
     &rmax_out_capac_norm(max_n_lok_zones,2),nbrkpt_ssm,mzone_puls_top,
     &mzone_puls_bot,no_puls_zones,iregjul1(max_n_oper_lines),
     &rmax_out_capac_wet(max_n_lok_zones,2),iclimate_opt,lsim_s65e_clim,
     &izone_bot_flood_flows_south,s65e_inflw_thres_high(max_n_seas),
     &eaa_conv_opt_reg(2,max_n_lok_zones,2),stgmxrg(max_n_wcas),
     &rmin_clim_indx_thres(max_n_lok_zones,2,2),
     &ipulse_level_in_zone(max_n_lok_zones),no_pred_storms_thres,
     &qplsl(max_n_puls_zones,max_n_puls_rel),
     &frac_depth_zone(max_n_lok_zones,max_n_seas),
     &avg_pulse_release(max_n_puls_zones,2),regstg1(max_n_oper_lines),
     &iopt_for_interp(max_n_lok_zones),iadd_ops_to_eaa_res_line,
     &qplscl(max_n_puls_zones,max_n_puls_rel),
     &stg_lok_sched(max_n_lok_zones,max_n_puls_rel),iflex_pulse_opt,
     &s65e_inflw_thres_low(max_n_seas),izone_bot_state_eaares,
     &igrav_sim_opt(max_n_eaa_condt,max_n_lok_zones),
     &cfsoth_litz,offset_reg_to_wcas(max_n_wcas),
     &no_of_pulse_rel(max_n_puls_zones),inorth_stor_opt,
     &iprop_res_prior_opt_lokreg,caloos_res_area,stlucie_res_area,
     &caloos_res_area_w_asr,rnorth_stor_to_lok,rlok_to_north_stor,
     &rnorth_stor_res_area,rmin_lok_stg_del_est,cfsoth_res,
     &itcns_res_opt,tayck_nubsl_res_area,s_forec_eaar_thres1,
     &totloketvol1(12),totloketvol3(12),s_forec_eaar_thres2,
     &totloketvol6(12),avg_demand_lok3(12),avg_demand_lok6(12),
     &no_of_res_reg_frm_lok(max_n_eaa_condt),cresbp_max_rate,
```

```
     &cal_res_dpth_thres_bp,lok_split_option,fract_mds_res,
     &fract_mds_litzone,stagelo_res,
     &itop_zone_indx(max_n_stor_w_oper_sch),
     &ibot_zone_indx(max_n_stor_w_oper_sch),
     &ialt_bot_zone_indx(max_n_stor_w_oper_sch),stlucie_res_max_dpth,
     &caloos_res_max_dpth,rmax_stl_res_bflw_cap,chgrnfistapoga(366),
     &chgdmdistapoga(366),chgdmdnls(366),chgrnfnls(366),
     &chgdmdnels(366),chgrnfnels(366),iregjul2(max_n_oper_lines),
     &ibot_zone_indx_s333reg(max_n_stor_w_oper_sch),
     &regstg2(max_n_oper_lines),ifloor_line_indx(max_n_stor_w_oper_sch),
     &opt_for_pulsing(max_n_lok_zones),opt_reg_lok_to_cal_res,
     &limit_reg_rel_glades(max_n_eaa_condt,max_n_lok_zones,2),
     &opt_reg_lok_to_stl_res,stor_area_name(max_n_stor_w_oper_sch)
C--------------------------------
      real iregjul,iregjul1,iregjul2
      character eaa_conv_opt_reg*4,opt_for_pulsing*7
     $,limit_reg_rel_glades*6,opt_reg_lok_to_cal_res*3
     $,opt_reg_lok_to_stl_res*3,stor_area_name*6
      logical lsim_s65e_clim
```

ovfllim.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: ovfllim.inc,v 1.4 2003/07/03 20:08:03 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ovfllim.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /ovfllim/ ndays_ovlflo_vol_limit(max_ncells,max_n_year,2),
     &excess_vol_ovlflo(max_ncells,max_n_year,2),
     &vol_ovlflo(max_ncells,max_n_year,2),
     &ratio_excess_ovlflo(max_ncells,max_n_year,2)
```

ovlnf.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: ovlnf.F,v 1.20 2005/06/10 13:29:15 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ovlnf.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE COMPUTES OVERLAND FLOW ,INFILTRATION,
C   AND EVAPOTRANSPIRATION FOR EACH GRID CELL. THIS
C   SUBROUTINE ALSO COMPUTES FLOW OVER PASSIVE WIERS
C   ACROSS PROTECTIVE LEVEES.
C
C   MAJOR VARIABLES ; OVERLAND FLOW CALCULATIONS
C
C    BSN       ARRAY OF BASIN INDICATORS
C    DETEN     DEPTH (FT) BELOW WHICH NO OVERLAND FLOW OCCURS
C    ELLS      LAND ELLEVATION AT EACH NODE (FT MSL)
C    ET        EVAPOTRANSPIRATION FROM ETCOMP SUBROUTINE
C    FILIM     CALCULATED SOIL INFILTRATION (FT)
C    FIMAX     INFILTRATION CAPACITY OF UNSATURATED SOIL
C    HM        AVG. OF UP + DOWNSTREAM STAGES
C    HU        STAGE OF UPSTREAM NODE
C    IT        ( 1 OR 2 ) DETERMINES DIRECTION OF FLOW CALCULATION
C    LUTYP     ARRAY OF LAND USE INDICATORS
C    NODE      SUBSCRIPT VALUE OF UPSTREAM NODE
C    NODE2         "       "     DOWNSTREAM  "
C    OFML      ARRAY OF MANNING'S VALUES FOR EACH LAND USE
C    OVMAX     MAXIMUM POSSIBLE OVERLAND FLOW VOLUME
C    POND      DEPTH OF SURFACE WATER AT EACH NODE (FT)
C    SINF      SOIL INFILTRATION RATES FOR EACH BASIN (FT/DAY)
C    VOLOF     VOLUME OF OVERLAND FLOW
C    WDTHOV    WIDTH OF FLOW PATH IN FEET (DX OR DY)
C    XLGTH     LENGTH "   "    "    "          "
C    ZN        MANNING'S ROUGHNESS COEFF.
C
C-------------------------------------------------------------------------------
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE ovlnf (jday, rain, depth_of_deficit, sumrain_unadj,
     & total_small_afact, total_afact, res_struc_inflw ,ofml2,
     & rmin_ofml2,allig_alley_res_fact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'et.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'budg.inc'
      INCLUDE 'wier.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'ovfllim.inc'

      DIMENSION node2_wr(5), rain(max_ncells),
     & res_struc_inflw(max_n_res), sumrain_unadj(max_ncells),
     & total_afact(max_ncells), total_small_afact(max_ncells),
     & vol_not_avail_for_flw(max_n_res), water_level(max_ncells),
     & ofml2(max_ncells),rmin_ofml2(max_ncells)

      INTEGER x, xw, y, y2, y3

      LOGICAL lthree_cells

      REAL land_surf_grad, mean_land_elev, mean_ponding,
     & mean_water_level

      DOUBLE PRECISION hu,hd,hc,hd_updated,hdx,hdy,avg_stg,
     & avg_stg_3cells,hgradm
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  initialize variables

      ETD = 0.0
      IF (day1) THEN
         DO node = 1,isum(maxy)
            pond_wr(node) = pond(node)
         END DO
      ENDIF
      DO i = 1,no_of_wiers
         DO k = 1,5
            QWIER(i,k) = 0.0
         END DO
      END DO
c
c  for cells containing a reservoir whose area is less than total area of grid
c  cell(s),and  reservoir is NOT an independent, separate entity, structural
c  inflow into reservoir is NOT available for overland flow (if applicable)
```

```
c  from reservoir cell(s) to non reservoir cell(s)
c  daily_inflow_to_res_frm_cnl(ia): structural inflow into reservoir ia
c
      DO ia = 1,ntotres
         IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &   (ia) .eq. 'NO ') THEN
            vol_not_avail_for_flw(ia) = daily_inflow_to_res_frm_cnl(ia)
     &        + res_struc_inflw(ia)
         ELSE
            vol_not_avail_for_flw(ia) = 0.0
         ENDIF
      END DO
C-----------------------------------------------------------------------
c  Time Loop for overland flow;NTSTEP_OV is number of time steps per day
c-----------------------------------------------------------------------
      DO 200 ITIME = 1,NTSTEP_OV
         NCA_OV = NCA_OV + 1
         NCA_ALT = NCA_ALT + 1
         IF (NCA_OV .GT. MAX_NCA_OV) NCA_OV = 1
         IF (NCA_ALT .GT. 2) NCA_ALT = 1
c
c
         GO TO (10,20), NCA_ALT
   10    JY = MAXY
         KY = 1
         INCR = -1
         GO TO 30
   20    JY = 1
         KY = MAXY
         INCR = 1
   30    DO 190 Y = JY,KY,INCR
            GO TO (40,50), NCA_ALT
   40       JX = MINX(Y)
            KX = MAXX(Y)
            INCR = 1
            GO TO 60
   50       JX = MAXX(Y)
            KX = MINX(Y)
            INCR = -1
   60       DO 190 X = JX,KX,INCR
               NODE = X - MINX(Y) +  1 + ISUM(Y)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c  impose boundary condition for overland flow on southwestern
c  part of ENP. Assume gradient of water level to be same as
C  land surface IF slope of land surface is downward toward
C  the west coast
c
               IF (y .GE. 4 .AND. y .LT. 26 .AND. x .EQ. minx(y)) THEN
                  IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                     y3 = y-1
                  ELSE
                     y3 = y+1
                  ENDIF
                  node_to_east = node + 1
                  IF (minx(y) .GE. minx(y3)) THEN
                     node_to_south = minx(y) - minx(y3) +  1 + isum(y3)
```

```
                  ELSE
                     node_to_south = node_to_east
                  ENDIF

                  mean_land_elev = (ells(node_to_east)+ells
     $            (node_to_south)) / 2.0
                  mean_ponding = (pond(node_to_east)+pond(node_to_south)
     $            ) / 2.0
                  mean_water_level = mean_land_elev + mean_ponding
                  land_surf_grad = mean_land_elev - ells(node)
                  IF (land_surf_grad .GT. 0.0) THEN
                     water_level(node) = mean_water_level -
     $               land_surf_grad
                     water_level(node) = amax1(water_level(node),ells
     $               (node) + deten(node))
                     pond(node) = water_level(node) - ells(node)
                  ENDIF
               ENDIF
c
               RINFLT = 0.0
C-----------------
C   CALCULATE HEAD DIFFERENCE(HU-HD) AND HYDRAULIC RADIUS (HM)
C-----------------
C
               pond_beg = pond(node)
C-----------------
C   CALCULATE HEAD DIFFERENCE(HU-HD) AND HYDRAULIC RADIUS (HM)
C-----------------
               LTHREE_CELLS = .FALSE.
               hu = pond(node) + ells(node)
               IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                  max_col = maxx(y)
                  y2 = y-1
                  IF (y .EQ. 1) y2 = 1
               ELSE
                  max_col = minx(y)
                  y2 = y+1
                  y2 = min(y2,maxy)
               ENDIF
               IF ((IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 10) .OR.
     $         IBSN(NODE) .EQ. 40 .OR. IBSN(NODE) .EQ. 42) GO TO 190
               IF (x .NE. max_col) THEN
                  IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                     node2_x = node + 1
                  ELSE
                     node2_x = node - 1
                  ENDIF
                  hdx = pond(node2_x) + ells(node2_x)
                  IF (x.LE.maxx(y2) .AND. x .GE. minx(y2)) THEN
                     node2_y = x - minx(y2) +  1+ isum(y2)
                     hdy = pond(node2_y) + ells(node2_y)
                     avg_stg_3cells = (hu + hdy + hdx) / 3.0
                     IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                        IF (X .GT. MXOV(Y) ) GO TO 190
                        IF (lutyp(node) .NE. 0) THEN
                           IF (ibsn(node) .EQ. ibsn(node2_x) .OR. (ibsn
     $                     (node) .EQ. 63 .OR. ibsn(node2_x) .EQ. 63)
```

```
     $                     .OR. ibsn(node2_x) .EQ. 44 .OR. ibsn(node2_x)
     $                     .EQ. 64 .OR. (ibsn(node) .EQ. 65 .OR. ibsn
     $                     (node2_x) .EQ. 65)) THEN
                              IF (ibsn(node) .EQ. ibsn(node2_y).OR.ibsn
     $                        (node) .EQ. 44 .OR. ibsn(node) .EQ. 64
     $                        .OR. ibsn(node2_y) .EQ. 64 .OR. ibsn(node)
     $                        .EQ. 65) THEN
                                 avg_stg = avg_stg_3cells
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                                 LTHREE_CELLS = .TRUE.
                              ELSE
                                 avg_stg = (hu + hdx) / 2.0
                              ENDIF
                           ELSE
                              IF (ibsn(node) .EQ. ibsn(node2_y).OR.ibsn
     $                        (node) .EQ.  44 .OR. ibsn(node) .EQ. 64
     $                        .OR. ibsn(node2_y) .EQ. 64 .OR. ibsn(node)
     $                        .EQ. 65) THEN
                                 avg_stg = (hu + hdy) / 2.0
                              ENDIF
                           ENDIF
                        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                     ELSE
                        XW = X - 1
                        XW = MAX(XW,MINX(Y))
                        IF (XW .GT. MXOV(Y)) GO TO 190
                        IF ((ibsn(node) .EQ. ibsn(node2_x) .OR. ibsn
     $                  (node) .EQ. 44 .OR. ibsn(node) .EQ. 64 .OR.
     $                  (ibsn(node) .EQ. 63 .OR. ibsn(node2_x) .EQ. 63
     $                  .OR. ibsn (node) .EQ. 65 .OR. ibsn(node2_x) .EQ.
     $                  65)) .AND. lutyp(node2_x) .NE. 0) THEN
                           IF ((ibsn(node) .EQ. ibsn(node2_y) .OR. ibsn
     $                     (node2_y) .EQ. 44 .OR. ibsn(node2_y) .EQ. 64
     $                     .OR. ibsn(node) .EQ. 64 .OR. ibsn(node2_y)
     $                     .EQ. 65) .AND. lutyp(node2_y) .NE. 0) THEN
                              avg_stg = avg_stg_3cells
                              LTHREE_CELLS = .TRUE.
                           ELSE
                              avg_stg = (hu+hdx)/2.0
                           ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                        ELSE
                           IF ((ibsn(node) .EQ. ibsn(node2_y) .OR. ibsn
     $                     (node2_y) .EQ. 44 .OR. ibsn(node2_y) .EQ.
     $                      64 .OR. ibsn(node) .EQ. 64 .OR. ibsn(node)
     $                     .EQ. 65) .AND. lutyp(node2_y) .NE. 0) THEN
                              avg_stg = (hu+hdy)/2.0
                           ENDIF
                        ENDIF
                     ENDIF
                  ELSE
                     avg_stg = (hu + hdx)/2.0
                  ENDIF
               ELSE
                  IF (x.LE.maxx(y2) .AND. x .GE. minx(y2)) THEN
                     node2_y = x - minx(y2) +  1+ isum(y2)
```

```
                     node2_x = node
                     hdy = pond(node2_y) + ells(node2_y)
                     hdx = hu
                     avg_stg = (hu + hdy)/2.0
                  ENDIF
               ENDIF
               IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                  IF (LUTYP(NODE) .EQ. 0) GO TO 190
               ENDIF
c
               IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 2) THEN
                  IT1 = 1
                  IT2 = 2
                  INCR_IT = 1
               ELSE
                  IT1 = 2
                  IT2 = 1
                  INCR_IT = -1
               ENDIF
c     IF ((x.EQ.31.AND.y.EQ.53).or.(x.eq.32.and.y.eq.53)) THEN
c      WRITE(445,*) nca_ov,IT1,IT2,INCR_IT,month,iday
c    +, ifyr+iyear-1,'  INCRCRCR  ',hu,hdx,hdy,avg_stg
c     ENDIF
               sfactor1 = 0.0
               sfactor2 = 0.0
               res_lgth1= 0.0
               res_lgth2= 0.0
               DO ir = 1,noresincell(node)
                  ia = ires_index_in_cell(node,ir)
                  IF (sfactor(ia) .LT. sfactmin .and. ires_small_sim
     &            (ia) .eq. 'YES') THEN
                     sfactor1 = sfactor(ia) + sfactor1
                     res_lgth1 = sqrt(res_area(ia)/640.) +
     &               res_lgth1
                  ENDIF
               ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------
               DO 180 IT = IT1,IT2,INCR_IT
C-----------------
C   CALCULATE FLOWS IN THE WEST-EAST DIRECTION THE FIRST ITERATION
C   AND IN THE NORTH-SOUTH DIRECTION IN THE SECOND ITERATION
C-----------------
C  WHEN BSN(NODE) DOES NOT EQUAL BSN(NODE2) THERE IS AN INTERVENING
C     LEVEE. THEREFORE DON'T CALCULATE OVERLAND FLOW.
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  GO TO (80,70) IT
   70             IF (X .GT. MAXX(Y2) .OR. X .LT. MINX(Y2)) GO TO 180
                  IF ((IBSN(NODE) .GE. 7 .AND. IBSN(NODE) .LE. 10) .OR.
     $            IBSN(NODE) .EQ. 40 .OR. IBSN(NODE) .EQ. 42) GO TO 160
                  NODE2 = NODE2_Y
                  IF (ibsn(node) .NE. 64 .AND. ibsn(node2) .NE. 64) THEN
                     IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                        IF (IBSN(NODE) .NE. IBSN(NODE2) .AND. IBSN(NODE)
     $                  .NE. 44 .AND. IBSN(NODE) .NE. 65) GO TO 160
                     ELSE
```

```
                        IF (LUTYP(NODE2) .EQ. 0) GO TO 180
                        IF (IBSN(NODE) .NE. IBSN(NODE2) .AND. IBSN
     $                  (NODE2) .NE. 44 .AND. IBSN(NODE2) .NE. 65)
     $                  GO TO 160
                     ENDIF
                  ENDIF
                  sfactor2 = 0.0
                  res_lgth2= 0.0
                  DO ir = 1,noresincell(node2)
                    ia = ires_index_in_cell(node2,ir)
                     IF (sfactor(ia) .LT. sfactmin .and. ires_small_sim
     $               (ia) .eq. 'YES') THEN
                        sfactor2 = sfactor(ia) + sfactor2
                        res_lgth2 = sqrt(res_area(ia)/640.) +
     $                  res_lgth2
                     ENDIF
                  ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  WDTHOV = (DY-res_lgth1)*sqrt(sfactor1)
     $            + (DY - res_lgth2)*sqrt(sfactor2)
     $            + DY * (1.0-sqrt(sfactor1)) + DY *(1.0-sqrt(sfactor2))
                  WDTHOV = WDTHOV / 2.0
                  XLGTH = DX
                  GO TO 90
   80             IF ( X .EQ. MAX_COL) GO TO 180
                  IF ((IBSN(NODE) .GE.  7 .AND. IBSN(NODE) .LE. 10) .OR.
     $            IBSN(NODE) .EQ. 40 .OR. IBSN(NODE) .EQ. 42) GO TO 160
                  NODE2 = NODE2_X
                  IF (IBSN(NODE) .NE. 63 .AND. IBSN(NODE2) .NE. 63 .AND.
     $            IBSN(NODE) .NE. 65 .AND. IBSN(NODE2) .NE. 65) THEN
                     IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                        IF (IBSN(NODE) .NE. IBSN(NODE2) .AND. IBSN
     $                  (NODE2) .NE. 44 .AND. ibsn(node2) .NE. 64)
     $                  GO TO 160
                     ELSE
                        IF (LUTYP(NODE2) .EQ. 0) GO TO 180
                        IF (IBSN(NODE) .NE. IBSN(NODE2) .AND. IBSN(NODE)
     $                  .NE. 44 .AND. ibsn(node) .NE. 64) GO TO 160
                     ENDIF
                  ENDIF
                  sfactor2 = 0.0
                  res_lgth2= 0.0
                  DO ir = 1,noresincell(node2)
                     ia = ires_index_in_cell(node2,ir)
                     IF (sfactor(ia) .LT. sfactmin .and. ires_small_sim
     $               (ia) .eq. 'YES') THEN
                        sfactor2 = sfactor(ia) + sfactor2
                        res_lgth2 = sqrt(res_area(ia)/640.) + res_lgth2
                     ENDIF
                  ENDDO
                  WDTHOV = (DX-res_lgth1)*sqrt(sfactor1)
     $            + (DX - res_lgth2)*sqrt(sfactor2)
     $            + DX * (1.0-sqrt(sfactor1)) + DX *(1.0-sqrt(sfactor2))
                  WDTHOV = WDTHOV / 2.0
                  XLGTH = DY
C-----------------
C   CALCULATE HEAD DIFFERENCE(HU-HD) AND HYDRAULIC RADIUS (HM)
```

```
C-----------------
C------------------------------
C    TEST IF NODE RECEIVING FLOW IS IN OCEAN
C    IF IT IS,THE HYDRAULIC RADIUS IS ASSUMED
C    TO BE THE PONDING DEPTH AT UPSTREAM CELL
C    IF HEAD AT CELL IS GREATER
C------------------------------
   90             IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                     node_c1 = node2
                     node_c2 = node
                     h_c = hu
                  ELSE
                     node_c1 = node
                     node_c2 = node2
                     h_c = ells(node_c2) + pond(node_c2)
                  ENDIF
                  HU_UPDATED = H(NODE) + POND(NODE)
                  IF (LUTYP(NODE_C1) .NE.  0 .OR. POND(NODE_C1)+ELLS
     $            (NODE_C1).GT.H_C) THEN
                     IF (LUTYP(NODE_C1) .EQ. 0) THEN
                        SWDPTH = H(NODE_C1) + POND(NODE_C1) - ELLS
     $                  (NODE_C2)
                        SWDPTH = AMAX1(SWDPTH,0.0)
                     ELSE
                        SWDPTH = POND(NODE2)
                     ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                     HM = (POND_BEG*(1.0-sfactor1)+SWDPTH*(1.0-sfactor2)
     $               )/(2.0-(sfactor1+sfactor2))
                  ELSE
                     HM = POND_BEG
                  ENDIF
                  IF (HM .LE. DETEN(NODE)) GO TO 160
                  HD = POND(NODE2) + ELLS(NODE2)
                  IF (node2 .EQ. node2_x) THEN
                     IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                        node2_pres = node2_y
                     ELSE
                        node2_pres = node
                     ENDIF
                     IF (ibsn(node2_y) .EQ. ibsn(node) .OR. ibsn
     $               (node2_pres) .EQ. 44 .OR. ibsn(node2_pres) .EQ.
     $               65 .OR. ibsn(node2_y) .EQ. 64 .OR. ibsn(node) .EQ.
     $               64) THEN
                        HC = POND(node2_y) + ELLS(node2_y)
                     ELSE
                        HC = HU
                     ENDIF
                     kdirect = 1
                  ELSEIF (node2 .EQ. node2_y) THEN
                     IF (x .EQ. max_col) THEN
                        node2_col = node
                     ELSE
                        node2_col = node2_x
                     ENDIF
                     IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
```

```
                      node2_pres = node2_col
                   ELSE
                      node2_pres = node
                   ENDIF
                   IF (ibsn(node2_col) .EQ. ibsn(node) .OR. ibsn
     $             (node2_pres) .EQ. 44 .OR. ibsn(node) .EQ. 63 .OR.
     $             ibsn(node2_col) .EQ.  63 .OR. ibsn(node) .EQ. 65
     $             .OR. ibsn(node2_col) .EQ. 65) THEN
                      HC = POND(node2_col) + ELLS(node2_col)
                   ELSE
                      HC = HU
                   ENDIF
                   kdirect = 2
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                ENDIF

                hgradm = max(abs(hu-hd),1.0E-4)
                COSN = 1.0 / SQRT(SQRT((HGRADM)**2 + (HU-HC)**2))
                IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                   IF (lutyp(node2) .EQ. 0) THEN
                      ZN = OFML2(NODE)*OFML_fact(LUTYP
     $                (NODE),kdirect) * HM ** OFML(LUTYP(NODE),2)
                      ZN = amax1(ZN,rmin_ofml2(NODE))
                   ELSE
                      ZN = (OFML2(NODE)*OFML_fact(LUTYP
     $                (NODE),kdirect)*HM**OFML(LUTYP(NODE),2) + OFML2
     $                (NODE2)*OFML_fact(LUTYP(NODE2)
     $                ,kdirect)*HM**OFML(LUTYP(NODE2),2))/2.0
                      ZN = amax1(ZN,rmin_ofml2(NODE))
                   ENDIF
                ELSE
                   IF (lutyp(node) .EQ. 0) THEN
                      ZN = OFML2(NODE2)*OFML_fact(LUTYP
     $                (NODE2),kdirect) * HM ** OFML(LUTYP(NODE2),2)
                      ZN = amax1(ZN,rmin_ofml2(NODE2))
                   ELSE
                      ZN = (OFML2(NODE)*OFML_fact(LUTYP
     $                (NODE),kdirect)*HM**OFML(LUTYP(NODE),2) + OFML2
     $                (NODE2)*OFML_fact(LUTYP(NODE2),
     $                kdirect)*HM**OFML(LUTYP(NODE2),2))/2.0
                      ZN = amax1(ZN,rmin_ofml2(NODE))
                   ENDIF
                ENDIF
                IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                  IF (y .eq. 36 .and. node2 .eq. node2_y .and.
     $            (ibsn(node) .eq. 3 .or. ibsn(node) .eq. 6)) THEN
                     ZN = ZN * allig_alley_res_fact
                  ENDIF
                ELSE
                  IF (y .eq. 35 .and. node2 .eq. node2_y .and.
     $            (ibsn(node) .eq. 3 .or. ibsn(node) .eq. 6)) THEN
                     ZN = ZN * allig_alley_res_fact
                  ENDIF
                ENDIF
                OFM = 1.49 / ZN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C----------------
```

```
C  CALCULATE VOLUME OF OVERLAND FLOW USING MANNING'S EQUATION
C  THEN MAKE SURE THIS AMOUNT DOES NOT CAUSE GRADIENT TO REVERSE SIGN.
C-----------------
                  IF (HU - HD) 100, 160, 130
  100             VOF = OFM * MAX(HD - HU,1.0E-4)/SQRT(XLGTH)*COSN * HM
     $            ** (2./3.)
                  VOLOV = VOF * DTS * WDTHOV * HM
c
                  IF (LTHREE_CELLS) THEN
                     IF (hdx .LE. avg_stg_3cells .OR. hdy .LE.
     $               avg_stg_3cells) THEN
                        OVMAX = max(hd - hu_updated,0.0) * GDAR
     $                  * (1.0-sfactor2-carfact(node2)) / 2.0
                     ELSE
                        OVMAX = (hd - avg_stg) * GDAR * (1.0-sfactor2
     $                  - carfact(node2))
                     ENDIF
                  ELSE
                     OVMAX = (hd - avg_stg) * GDAR * (1.0-sfactor2
     $                  - carfact(node2))
                  ENDIF
c
                  OVTEST = (POND(NODE2) - DETEN(NODE2)) * GDAR
     $            * (1.0 - sfactor2 - carfact(node2))
                  OVTEST = AMAX1(OVTEST,0.0)
                  IF (OVMAX .GT. OVTEST) OVMAX = OVTEST
c     IF (ifyr+iyear-1.ge.1971) THEN
c     IF ((x.EQ.31.AND.y.EQ.53).or.(x.eq.32.and.y.eq.53)) THEN
c      WRITE(445,*) OVMAX/43560.,VOLOV/43560.,hu,hu_updated,hd
c    +, avg_stg,avg_stg_3cells,node,node2,month,iday,'  AAA  '
c    +, ibsn(node),ibsn(node2),OFM,HM,WDTHOV,COSN,month,iday
c    +, ifyr+iyear-1
c     ENDIF
c     ENDIF
                  IF (VOLOV - OVMAX) 110, 110, 120
  110             POND(NODE2) = POND(NODE2) - VOLOV / (GDAR*
     $            (1.0-sfactor2-carfact(node2)))
                  POND (NODE) = POND(NODE) + VOLOV / (GDAR*
     $            (1.0-sfactor1-carfact(node)))
c     IF (x.EQ.28.AND.y.EQ.53)THEN
c      PRINT *,ells(node)+pond(node),ells(node2)+pond(node2),' aft flow
c    $ cccc ',sfactor1,sfactor2,ibsn(node),ibsn(node2)
c     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  vol_ovlflo(node,iyear,it) = vol_ovlflo(node,iyear,it)
     $            + VOLOV/(43560. * 1000.)
C
                  IF ( LP(20) ) THEN
                     IF (.NOT. DAY1) THEN
                        IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                           OVLFLO (NODE,IT) = OVLFLO(NODE,IT) - VOLOV
                           OVLFLO_daily(NODE,IT) = - VOLOV +
     $                     OVLFLO_daily(NODE,IT)
                        ELSE
                           OVLFLO (NODE2,IT) = OVLFLO(NODE2,IT) + VOLOV
                           OVLFLO_daily(NODE2,IT) = VOLOV + OVLFLO_daily
     $                     (NODE2,IT)
```

```
                      ENDIF
                   ENDIF
                ENDIF
C
                GO TO 160
  120           POND(NODE2) = POND(NODE2) - OVMAX / (GDAR
     $          *(1.0-sfactor2-carfact(node2)))
                POND(NODE) = POND(NODE) + OVMAX / (GDAR
     $          *(1.0-sfactor1-carfact(node)))
c     IF (x.EQ.28.AND.y.EQ.53)THEN
c      PRINT *,ells(node)+pond(node),ells(node2)+pond(node2),' aft flow
c    $ dddd ',sfactor1,sfactor2
c     ENDIF
                vol_ovlflo(node,iyear,it) = vol_ovlflo(node,iyear,it)
     $          + VOLOV / (43560. * 1000.)
                IF (OVMAX .LT. OVTEST) THEN
                   ndays_ovlflo_vol_limit(node,iyear,it) =
     $             ndays_ovlflo_vol_limit(node,iyear,it) + 1
                   excess_vol_ovlflo(node,iyear,it) =
     $             excess_vol_ovlflo(node,iyear,it) + (VOLOV-OVMAX)
     $             / (43560. * 1000.)
                ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
                IF ( LP(20) ) THEN
                   IF ( .NOT. DAY1) THEN
                      IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                         OVLFLO (NODE,IT) = OVLFLO (NODE,IT) - OVMAX
                         OVLFLO_daily(NODE,IT) = - OVMAX +
     $                   OVLFLO_daily(NODE,IT)
                      ELSE
                         OVLFLO (NODE2,IT) = OVLFLO (NODE2,IT) + OVMAX
                         OVLFLO_daily(NODE2,IT) = OVMAX + OVLFLO_daily
     $                   (NODE2,IT)
                      ENDIF
                   ENDIF
                ENDIF
C
                GO TO 160
C-----------------
  130           VOF = OFM * MAX(HU - HD,1.0E-4)/SQRT(XLGTH)*COSN * HM
     $          ** (2./3.)
                VOLOV = VOF * DTS * WDTHOV * HM
                OVMAX = (avg_stg - HD) * GDAR * (1.0 - sfactor1
     $           - carfact(node))
c
                IF (LTHREE_CELLS) THEN
                   IF ((hdx .GT. hu .OR. hdy .GT. hu) .OR.
     $             avg_stg_3cells .LE. hdx .OR. avg_stg_3cells .LE.
     $             hdy) THEN
                      OVMAX = max(hu_updated - hd,0.0) * gdar
     $                * (1.0 - sfactor1 - carfact(node)) / 2.0
                   ENDIF
                ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                OVTEST = (POND(NODE) - DETEN(NODE)) * GDAR
```

```
     $            * (1.0 - sfactor1 - carfact(node))
                  OVTEST = AMAX1(OVTEST,0.0)
                  IF (OVMAX .GT. OVTEST) OVMAX = OVTEST
c     IF (ifyr+iyear-1.ge.1971) THEN
c     IF ((x.EQ.31.AND.y.EQ.53).or.(x.eq.32.and.y.eq.53)) THEN
c      WRITE(445,*) OVMAX/43560.,VOLOV/43560.,hu,hu_updated,hd
c    +, avg_stg,avg_stg_3cells,node,node2,month,iday,'  BBB  '
c    +, ibsn(node),ibsn(node2),OFM,HM,WDTHOV,COSN,month,iday
c    +, ifyr+iyear-1
c     ENDIF
c     ENDIF
c     IF (x.EQ.28.AND.y.EQ.53)THEN
c      PRINT *,OVMAX/43560.,VOLOV/43560.,hu,hu_updated,hd
c    +, avg_stg,avg_stg_3cells,node,node2,month,iday,'  BBB  '
c    +, sfactor1,sfactor2,OFM,HM,WDTHOV,COSN
c     ENDIF
                  IF (VOLOV - OVMAX) 140, 140, 150
  140             POND(NODE2) = POND(NODE2) + VOLOV / (GDAR
     $            * (1.0-sfactor2-carfact(node2)))
                  POND (NODE) = POND(NODE) - VOLOV / (GDAR
     $            * (1.0-sfactor1-carfact(node)))
c     IF (x.EQ.28.AND.y.EQ.53)THEN
c      PRINT *,ells(node)+pond(node),ells(node2)+pond(node2),' aft flow
c    $  eeee  ',sfactor1,sfactor2
c     ENDIF
                  vol_ovlflo(node,iyear,it) = vol_ovlflo(node,iyear,it)
     $            + VOLOV / (43560. * 1000.)
C
                  IF ( LP(20) ) THEN
                     IF (.NOT. DAY1) THEN
                        IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                           OVLFLO ( NODE,IT) = OVLFLO ( NODE,IT) + VOLOV
                           OVLFLO_daily(NODE,IT) = VOLOV + OVLFLO_daily
     $                     (NODE,IT)
                        ELSE
                           OVLFLO ( NODE2,IT) = OVLFLO ( NODE2,IT) -
     $                     VOLOV
                           OVLFLO_daily(NODE2,IT) = - VOLOV +
     $                     OVLFLO_daily(NODE2,IT)
                        ENDIF
                     ENDIF
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
                  GO TO 160
  150             POND(NODE2) = POND(NODE2) + OVMAX / (GDAR
     $            * (1.0-sfactor2-carfact(node2)))
                  POND(NODE) = POND(NODE) - OVMAX / (GDAR
     $            * (1.0-sfactor1-carfact(node)))
c     IF (x.EQ.28.AND.y.EQ.53)THEN
c      PRINT *,ells(node)+pond(node),ells(node2)+pond(node2),' aft flow
c    $  ffff  ',sfactor1,sfactor2
c     ENDIF
                  vol_ovlflo(node,iyear,it) = vol_ovlflo(node,iyear,it)
     $            + VOLOV / (43560. * 1000.)
                  IF (OVMAX .LT. OVTEST) THEN
                     ndays_ovlflo_vol_limit(node,iyear,it) =
```

```
     $               ndays_ovlflo_vol_limit(node,iyear,it) + 1
                     excess_vol_ovlflo(node,iyear,it) =
     $               excess_vol_ovlflo(node,iyear,it) + (VOLOV-OVMAX)
     $               / (43560. * 1000.)
                  ENDIF
C
                  IF ( LP(20) ) THEN
                     IF (.NOT. DAY1) THEN
                        IF (nca_ov .EQ. 1 .OR. nca_ov .EQ. 3) THEN
                           OVLFLO (NODE,IT) = OVLFLO (NODE,IT) + OVMAX
                           OVLFLO_daily(NODE,IT) = OVMAX + OVLFLO_daily
     $                     (NODE,IT)
                        ELSE
                           OVLFLO (NODE2,IT) = OVLFLO (NODE2,IT) - OVMAX
                           OVLFLO_daily(NODE2,IT) = - OVMAX +
     $                     OVLFLO_daily(NODE2,IT)
                        ENDIF
                     ENDIF
                  ENDIF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
  160             IF (iwier_index(node) .NE. 0) THEN
                     IF (IT .EQ. IT2) THEN
                        ndirection = 0
                        IF (wier_orient(iwier_index(node),1) .EQ. 'E')
     $                  THEN
                           ndirection = ndirection + 1
                           NODE2_WR(ndirection) = NODE + 1
                        ELSE IF (wier_orient(iwier_index(node),2) .EQ.
     $                  'W') THEN
                           ndirection = ndirection + 1
                           NODE2_WR(ndirection) = NODE - 1
                        ELSE IF (wier_orient(iwier_index(node),3) .EQ.
     $                  'N') THEN
                           ndirection = ndirection + 1
                           NODE2_WR(ndirection) = X - MINX(Y+1) + 1 +
     $                     ISUM(Y+1)
                        ELSE IF (wier_orient(iwier_index(node),4) .EQ.
     $                  'S') THEN
                           ndirection = ndirection + 1
                           NODE2_WR(ndirection) = X - MINX(Y-1) +  1 +
     $                     ISUM(Y-1)
                        ELSE IF (wier_orient(iwier_index(node),5) .EQ.
     $                  'D') THEN
                           ndirection = ndirection + 1
                           NODE2_WR(ndirection) = node_weir_tw_constrnt
     $                     (iwier_index(node) ,ndirection)
                        ENDIF
                        DO idir = 1,ndirection
                           afact1 = 0.0
                           afact2 = 0.0
                           HW_WR = ELLS(NODE) + POND(NODE)
                           TW_WR = ELLS(NODE2_WR(idir)) + POND(NODE2_WR
     $                     (idir))
c
                           DO ir = 1,noresincell(NODE2_WR(idir))
                              ia = ires_index_in_cell(NODE2_WR(idir),ir)
```

```
                              ia_node2 = ia
                              afact2 = sfactor(ia)+afact2
                              TW_WR = ELLS(NODE2_WR(idir)) + POND_WR
     $                        (NODE2_WR(idir)) -
     $                        vol_not_avail_for_flw(ia) * 86400. /
     $                        (nnodes(ia)*(1.0 - afact2) * GDAR)
                           ENDDO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                           DO ir = 1,noresincell(node)
                              ia = ires_index_in_cell(node,ir)
                              afact1 = sfactor(ia)+afact1
                              HW_WR = ELLS(NODE) + POND_WR(NODE) -
     $                        vol_not_avail_for_flw(ia) * 86400. /
     $                        (nnodes(ia)*(1.0 - afact1) * GDAR)
                           END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                           IF (noresincell(NODE2_WR(idir)) .eq. 1
     $                     .AND. noresincell(node) .eq. 1 .AND.
     $                     ia_node2 .eq. ia) THEN
                              TW_WR = ELLS(NODE2_WR(idir))
     $                        + POND(NODE2_WR(idir))
                              HW_WR = ELLS(NODE) + POND(NODE)
                           ENDIF
                           TW_MAX = HW_WR * 0.50 + 0.50 * TW_WR
                           IF (tw_wr .LT. crstelev_w(iwier_index(node)))
     $                     THEN
                              TOTAL_AVAIL_DPTH = AMAX1(hw_wr-crstelev_w
     $                        (iwier_index(node)) ,0.0)
                              AVAIL_DPTH = amax1(TW_MAX - tw_wr,0.0)
                              TOTAL_AVAIL_DPTH = amin1(TOTAL_AVAIL_DPTH,
     $                        AVAIL_DPTH)
                           ELSE
                              IF (hw_wr .LT. crstelev_w(iwier_index
     $                        (node))) THEN
                                 TOTAL_AVAIL_DPTH = AMAX1(tw_wr
     $                           -crstelev_w (iwier_index(node)),0.0)
                                 AVAIL_DPTH = amax1(TW_MAX - hw_wr,0.0)
                                 TOTAL_AVAIL_DPTH = amin1
     $                           (TOTAL_AVAIL_DPTH,AVAIL_DPTH)
                              ELSE
                                 IF (hw_wr .GT. TW_WR) THEN
                                    TOTAL_AVAIL_DPTH = TW_MAX - tw_wr
                                 ELSE
                                    TOTAL_AVAIL_DPTH = TW_MAX - hw_wr
                                 ENDIF
                              ENDIF
                           ENDIF
                           AVVOL_W = TOTAL_AVAIL_DPTH * GDAR
                           tw_for_constrnt= ells(node_weir_tw_constrnt
     $                     (iwier_index(node) ,idir))+ pond
     $                     (node_weir_tw_constrnt(iwier_index(node),
     $                     idir))
c                          IF (tw_for_constrnt .LE. rmax_tw_stage_wier
c    +                     (iwier_index(node),idir)) THEN
                           IF (tw_wr .LE. crstelev_w(iwier_index
     $                     (node))) THEN
                              IF (HW_WR .GT. crstelev_w(iwier_index
```

```
     $                        (node)) .and. tw_for_constrnt
     $                        .LE. rmax_tw_stage_wier(iwier_index
     $                        (node),idir)) THEN
                                 STRUC_CAP=dcoef_wier(iwier_index
     $                           (node))*(HW_WR-crstelev_w
     $                           (iwier_index(node)))**PWR_WIER
     $                           (iwier_index(node))
                              ELSE
                                 STRUC_CAP = 0.0
                              ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                              SGN = 1
                           ELSE
                              IF (hw_wr .LT. crstelev_w(iwier_index
     $                        (node))) THEN
                                 STRUC_CAP=dcoef_wier(iwier_index
     $                           (node))*(TW_WR-crstelev_w
     $                           (iwier_index(node)))**PWR_WIER
     $                           (iwier_index(node))
                                 SGN = -1
                              ELSE
                                 IF (HW_WR .GT. TW_WR) THEN
                                   IF (tw_for_constrnt.LE.
     $                              rmax_tw_stage_wier(iwier_index
     $                              (node),idir)) THEN
                                    submerg_fact = (TW_WR - crstelev_w
     $                              (iwier_index(node))) / (HW_WR -
     $                              crstelev_w(iwier_index(node)))
                                    rel_crst_length = (HW_WR -
     $                              crstelev_w(iwier_index(node)))
     $                              /(HW_WR - crstelev_w(iwier_index
     $                              (node)) + 20.0)
                                    rmod_limit = 0.85 - 0.50
     $                              *rel_crst_length
                                    reduc_fact = (submerg_fact -
     $                              rmod_limit)/(1.0 -rmod_limit)
                                    reduc_fact = amax1(reduc_fact,0.0)
                                    reduc_fact = (1.0 - reduc_fact) **
     $                              0.143
                                    STRUC_CAP=dcoef_wier(iwier_index
     $                              (node)) * (HW_WR- crstelev_w
     $                              (iwier_index(node)))**PWR_WIER
     $                              (iwier_index(node)) *reduc_fact
                                   ELSE
                                    STRUC_CAP=0.0
                                   ENDIF
                                   SGN = 1
                                 ELSE
                                    submerg_fact = (HW_WR - crstelev_w
     $                              (iwier_index(node))) / (TW_WR -
     $                              crstelev_w(iwier_index(node)))
                                    rel_crst_length = (TW_WR -
     $                              crstelev_w(iwier_index(node)))
     $                              /(TW_WR - crstelev_w(iwier_index
     $                              (node)) + 20.0)
                                    rmod_limit = 0.85 - 0.50
     $                              *rel_crst_length
```

```
                                 reduc_fact = (submerg_fact -
     $                           rmod_limit)/(1.0 -rmod_limit)
                                 reduc_fact = amax1(reduc_fact,0.0)
                                 reduc_fact = (1.0 - reduc_fact) **
     $                           0.143
                                 STRUC_CAP=dcoef_wier(iwier_index
     $                           (node)) * (TW_WR- crstelev_w
     $                           (iwier_index(node)))**PWR_WIER
     $                           (iwier_index(node)) * reduc_fact
                                 SGN = -1
                              ENDIF
                           ENDIF
                        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c                       ELSE
c                          STRUC_CAP = 0.0
c                       ENDIF
                        STRUC_CAP = STRUC_CAP * 86400. / NTSTEP_OV
                        IF (lp(12)) THEN
                           IF (HW_WR .GT. TW_WR) THEN
                              WRITE(130,170) ifyr+iyear-1,month,iday,
     $                        itime,hw_wr,tw_wr ,crstelev_w
     $                        (iwier_index(node)),TOTAL_AVAIL_DPTH ,
     $                        STRUC_CAP*NTSTEP_OV / 86400.,AVVOL_W
     $                        / 86400.,AMIN1(STRUC_CAP*NTSTEP_OV,
     $                        AVVOL_W) / 86400. ,
     $                        tw_for_constrnt ,rmax_tw_stage_wier
     $                        (iwier_index(node),idir) ,weir_name_wca
     $                        (iwier_index(node),idir)
                           ELSE
                              WRITE(130,170) ifyr+iyear-1,month,iday,
     $                        itime,tw_wr,hw_wr ,crstelev_w
     $                        (iwier_index(node)),TOTAL_AVAIL_DPTH ,
     $                        STRUC_CAP*NTSTEP_OV / 86400.,AVVOL_W
     $                        / 86400.,AMIN1(STRUC_CAP*NTSTEP_OV,
     $                        AVVOL_W) / 86400. ,
     $                        tw_for_constrnt ,rmax_tw_stage_wier
     $                        (iwier_index(node),idir) ,weir_name_wca
     $                        (iwier_index(node),idir)
                           ENDIF
                        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                        QWIER(iwier_index(node),idir)=SGN*AMIN1
     $                  (STRUC_CAP,AVVOL_W) / 86400. + QWIER
     $                  (iwier_index(node),idir)
                        POND(node) = POND(node) - SGN * AMIN1
     $                  (STRUC_CAP,AVVOL_W)/GDAR
                        POND(node2_wr(idir)) = POND(node2_wr(idir)) +
     $                  SGN*AMIN1(STRUC_CAP,AVVOL_W) / GDAR
                        pond_adj_due_to_res(node) =
     $                  pond_adj_due_to_res(node) - SGN * (afact1
     $                  /(1.0 - afact1))*AMIN1(STRUC_CAP,AVVOL_W)
     $                  /GDAR
                        pond_adj_due_to_res(node2_wr(idir)) =
     $                  pond_adj_due_to_res(node2_wr(idir)) + SGN *
     $                  (afact2/(1.0 - afact2))*AMIN1(STRUC_CAP,
     $                  AVVOL_W)/GDAR
```

```
                           POND_WR(node) = POND(node) +
     $                     pond_adj_due_to_res(node)
                           POND_WR(node2_wr(idir)) = POND(node2_wr(idir)
     $                     ) + pond_adj_due_to_res(node2_wr(idir))
                           daily_chg_dpth_ovl_vol(NODE) =
     $                     daily_chg_dpth_ovl_vol(NODE) - SGN * AMIN1
     $                     (STRUC_CAP,AVVOL_W)/GDAR
                           daily_chg_dpth_ovl_vol(NODE2_wr(idir)) =
     $                     daily_chg_dpth_ovl_vol (node2_wr(idir)) + SGN
     $                     * AMIN1(STRUC_CAP,AVVOL_W) / GDAR
                        END DO
                     ENDIF
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
  180          CONTINUE
cbw DO 190 Y is nested with DO 190 X
  190    CONTINUE
  200 CONTINUE
C
      DO Y = MAXY,1,-1
         DO X = MINX(Y),MAXX(Y)
         NODE = X - MINX(Y) + 1 + ISUM(Y)
         IF (LUTYP(NODE) .NE. 0) THEN
c
c  calculate overland flow correction for estimate of
c    stage in reservoir whose outflow is going into
c    same cell in which reservoir is located and overland
c    flow is allowed to/from cell
c
            DO ir = 1,noresincell(node)
               ia = ires_index_in_cell(node,ir)
               IF (sfactor(ia).GE.sfactmin.OR.ires_small_sim
     $         (ia) .eq. 'NO ') THEN
                  dpth_of_flw = rain(node) /12. + qoutflw(ia,1)* 86400.
     $            /gdar - daily_ET(node) - daily_seep_loss_res(node) +
     $            chg_depth_levee_spg(node)
c                 depth_to_wt = amax1(ells(node)-stage_res(node,ir),0.0)
c    +            *s(node)
                  pot_ovl_adj = amax1(stage_res(node,ir)-ells(node),0.0)
     $            + dpth_of_flw
                  IF (daily_chg_dpth_ovl_vol(NODE) .LE. 0.0) THEN
                     sgn = -1.0
                  ELSE
                     sgn = 1.0
                  ENDIF
                  pot_ovl_adj = amax1(pot_ovl_adj,0.0)
c          IF (depth_to_wt .GT. 0.00001) THEN
c           daily_chg_dpth_ovl_vol1(node)=amin1(daily_chg_dpth_ovl_vol1
c    $         (node),depth_to_wt)
c          ELSE
                  daily_chg_dpth_ovl_vol1(node)= 0.0
c          ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  sum_chg_dpth_ovl_flw(NODE) = sum_chg_dpth_ovl_flw
     $            (NODE) + sgn*amin1(sgn*daily_chg_dpth_ovl_vol(NODE),
     $            pot_ovl_adj)
                  daily_chg_dpth_ovl_vol(node) = sgn*amin1(sgn
```

```
     $            *daily_chg_dpth_ovl_vol(NODE),pot_ovl_adj)
               ENDIF
            END DO
            IF (((IBSN(NODE) .LT. 7 .OR. IBSN(NODE) .GT. 10 .OR.
     $      land_use_type(LUTYP(NODE)) .EQ. 'WETLAND').AND.IBSN(NODE)
     $      .NE. 40 .AND. (IBSN(NODE) .NE. 42 .OR. land_use_type
     $      (LUTYP(NODE)).EQ. 'WETLAND')).OR. (runmode .EQ.'CALIB'
     $      .AND. IBSN(NODE) .NE. 10 .AND. IBSN(NODE) .NE. 40)) THEN
c
c note: for EAA, DO infiltration and et computations
c       in AGAREA SUBROUTINE
c therefore, non-EAA cell
               et_unsat_actual(node) = 0.0
               et_unsat_unacct(node) = 0.0
C
               IF (use_lec_et * lec_node_in_model(node) .EQ. 0 .OR. ibsn
     $         (node) .EQ. 40 .OR. frac_irr_area_t(node) .EQ. 0.) THEN
C
c*** original algorithm for non-eaa cell
c calculate soil infitration and et assuming no unsaturated zone
                  DPTHGW = (ELLS(node) - H(node)) * S(node)
                  IF (POND(node) .GT. DPTHGW) THEN
                     stage_in_cell = POND(node) - DPTHGW + ELLS(node)
                  ELSE
                     stage_in_cell = (POND(node)-DPTHGW)/S(node) + ELLS
     $               (node)
                  ENDIF
C---------------------
C     CALCULATE SOIL INFILTRATION
C---------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  INFILT(NODE) = 0.0
                  PERC(NODE) = 0.0
                  DO ir = 1,noresincell(node)
                     iresnum = ires_index_in_cell(node,ir)
                     IF (sfactor(iresnum).LT.sfactmin.and.ires_small_sim
     $               (iresnum) .eq. 'YES') THEN
c                       if (resname(iresnum) .eq. 'ENPBUF') print *
c    +                  ,pond_in_res(node,ir),ells_in_res(node,ir)
c    +                  ,'  LSETLSET  ',node
                        call etcomp(-1,ET,NODE,LUTYP_IN_RES(NODE,ir),
     $                  jday,depth_of_deficit,stage_res ,ET_outside_res,
     $                  iresnum,ir,sfactor(iresnum),ells_in_res,1)
c                       if (resname(iresnum) .eq. 'ENPBUF') print *
c    +                  ,pond_in_res(node,ir),et,ells_in_res(node,ir)
c    +                  ,'  LSETLSET2'
                        IF (pond_in_res(node,ir) .GT. ET) then
                           pond_in_res(node,ir) = pond_in_res(node,ir)
     $                     - ET
                           stage_res(node,ir) = ells_in_res(node,ir) +
     $                     pond_in_res(node,ir)
                        ELSE
                           stage_res(node,ir) = stage_res(node,ir)
     $                     - (ET - pond_in_res(node,ir))/s(node)
     $                     - pond_in_res(node,ir)
                           pond_in_res(node,ir) = 0.0
                        ENDIF
```

```
                     ENDIF
                  ENDDO
                  IF (POND(NODE) .GT. 0.) THEN
                     FILIM = AMIN1(SINF(NODE)*DT ,POND(NODE))
                     FIMAX = (ELLS(NODE) - H(NODE)) * S(NODE)
                     RINFLT = AMIN1(FIMAX,FILIM)
                     INFILT(NODE) = RINFLT * (1.0-carfact(NODE)
     $               - total_small_afact(NODE))
                     PERC(NODE) = INFILT(NODE)
                     POND(NODE) = POND(NODE) - RINFLT
c without unsaturated zone: all of RINFLT percolates,
c    therefore compute recharge now
                     RCHG(NODE) = RCHG(NODE) + RINFLT
                  ENDIF
C-----------------
C   CALCULATE THE ET VALUE FROM EACH NODE
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  if (total_afact(node)  .gt. 0.000001) then
                   depth_of_deficit = SUMET_outside_res(node) +
     $             seep_loss_res_cell(NODE) - SUMRAIN_unadj(node)/12.0 -
     $             sum_chg_depth_levee_spg_unadj(NODE) -
     $             (outf_fc_frm_res_to_cell(NODE) +
     $             daily_outf_fc_frm_res_to_cell(NODE) +
     $             sum_chg_dpth_ovl_flw(NODE))/total_afact(node)
                  else
                   depth_of_deficit = SUMET_outside_res(node) +
     $             seep_loss_res_cell(NODE) - SUMRAIN_unadj(node)/12.0 -
     $             sum_chg_depth_levee_spg_unadj(NODE) -
     $             (outf_fc_frm_res_to_cell(NODE) +
     $             daily_outf_fc_frm_res_to_cell(NODE) +
     $             sum_chg_dpth_ovl_flw(NODE))
                  endif
                  depth_of_deficit = amin1(depth_of_deficit,0.0)
                  IF (POND(node) .GT. 0.00001) THEN
                     est_cell_stg_o_res(node) = ells(node) -
     $               depth_of_deficit
                  ELSE
                     est_cell_stg_o_res(node) = stage_in_cell
                  ENDIF
c           IF (ibsn(node).EQ.27)
c    $      PRINT *,depth_of_deficit,SUMET_outside_res(node)
c    $       ,seep_loss_res_cell(NODE),SUMRAIN_unadj(node)/12.0
c    $       ,sum_chg_depth_levee_spg_unadj(NODE)
c    $       ,outf_fc_frm_res_to_cell(NODE)
c    $       ,daily_outf_fc_frm_res_to_cell(NODE)
c    $       ,sum_chg_dpth_ovl_flw(NODE),'  DEPTH OF DEFIC  '
c    $       ,est_cell_stg_o_res(node)
c                 depth_of_deficit = amax1(depth_of_deficit,0.0)
                  CALL ETCOMP (1,ET,NODE,LUTYP(NODE),JDAY ,
     $            depth_of_deficit,stage_res ,ET_outside_res,
     $            iresnum,1,total_small_afact(node),ells_in_res,1)
                  et_cnl=kmax(lutyp(node))*potet(ietzon(node))/12.0
                  IF (.NOT. DAY1) THEN
                     ETM(NODE) = ETM(NODE) + ET*(1.0-
     $               total_small_afact(node)-carfact(node))
     $               + et_cnl * carfact(node)
```

```
                  ENDIF
                  IF (POND(NODE) .GT. 0.00001) THEN
                     SUMET(NODE) = SUMET(NODE) + ET
                     SUMET_outside_res(NODE) = SUMET_outside_res(NODE) +
     $               ET_outside_res
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  daily_ET(NODE) = ET
                  daily_ET_outside_res(NODE) = ET_outside_res
                  ETD = POND(NODE) - ET
                  IF ( ETD ) 210,220,230
  210             RCHG(NODE) = RCHG(NODE) + ETD
  220             POND(NODE) = 0.0
                  GO TO 240
  230             POND(NODE) = ETD
  240             CONTINUE
C
               ELSE
c*** revised algorithm for non-eaa cell
c calculate soil infitration and et assuming unsaturated zone exists
c If we're in the LECDA and the LEC ET routine is turned on, DO this section.
c Subr. lec_et_comp computes et stuff, infiltration, ponding, recharge, etc.
C---------------------
C   CALCULATE SOIL INFILTRATION
C---------------------
c
                  rinflt = 0.0
                  infilt(node) = 0.0
                  perc(node) = 0.0
                  IF (pond(node) .GT. 0.) THEN
                     filim = amin1(sinf(node)*dt ,pond(node))
                     fimax = (ells(node) - h(node)) * s(node) - solmx
     $               (node)
                     fimax = amax1(fimax,0.0)
                     rinflt = amin1(fimax,filim)
                     pond(node) = pond(node) - rinflt
                  ENDIF
                  infilt(node) = rinflt*(1.0-total_small_afact(node)
     $            - carfact(node))
c update rinflt:
c   rinflt = natural infiltration + net irrigation for the node
                  rinflt = rinflt + supply_t(node)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------
C   CALCULATE THE ET VALUE FROM EACH NODE
c with unsaturated zone: not all of RINFLT percolates,
c   therefore compute recharge within lec_et_comp SUBROUTINE
c note: rinflt and rchg(node) are not identical,
c       rinflt is affected by available ponding, infiltration rates,
c       available void space in uz, nd irrigation pumpage while rchg()
c       is influenced by percolation, pumpage (ws + irrigation) and ET
c       (ponding/sat and unaccounted for unsat)
                  CALL lec_et_comp(node, pond(node), rinflt, rchg(node),
     $            lutyp(node), h(node), drz(lutyp(node)), srz(lutyp
     $            (node)), ells(node), ibsn(node),ietzon(node),adjcff
     $            (month),month,jday, iday,total_small_afact)
                  et_cnl=kmax(lutyp(node))*potet(ietzon(node))/12.0
```

```
                  IF (.NOT. DAY1) THEN
                     ETM(NODE) = ETM(NODE) + (et_sat(node) + et_ponding
     $               (node))*(1.0-carfact(node)) + et_unsat_actual(node)
     $               *(1.0-carfact(node)) + et_cnl * carfact(node)
                  ENDIF
                  ETD = POND(NODE) - (et_sat(node) + et_ponding(node))
                  IF (POND(NODE) .GT. 0.0001) SUMET(NODE) = SUMET(NODE)
     $            + et_sat(node) + et_ponding(node)
                  daily_ET(NODE) = et_sat(node) + et_ponding(node)
                  IF ( ETD ) 250,260,270
  250             RCHG(NODE) = RCHG(NODE) + ETD
  260             POND(NODE) = 0.0
                  GO TO 280
  270             POND(NODE) = ETD
  280             CONTINUE
                  et_sat(node) = et_sat(node)*(1.0-carfact(node))
                  et_ponding(node) = et_ponding(node)*(1.0 -
     $            carfact(node)) + et_cnl * carfact(node)
               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            ENDIF
         ENDIF
      END DO
      END DO
c
      DO NODE = 1,max_ncells
      IF (lutyp(NODE) .EQ. 0) THEN
         perc(node)   = 0.0
         infilt(node) = 0.0
      ENDIF
      END DO
C-----------------
      RETURN
  170 FORMAT(i5,3i3,2x,4f7.2,2x,3f7.0,2f7.2,3x,A6)
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

printlp.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: printlp.F,v 1.15 2005/06/10 13:29:16 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/printlp.F,v $
C   $Name: rel-5-5 $
C
C
C   PRINT OUT THE DESIRED ARRAYS WHEN CALLED BY MAIN PROGRAM
C   PRINTING OCCURS WHEN LP INDEX IS TRUE
C   VARIABLES:
C    AGP       TOTAL PUMPAGE IN AG AREA
C    B         CHAR. STRING TO BLANK FILL WHEN MINX # 1
C    IDD       DAY OF MONTH
C    IDYMAX    NUMBER OF DAYS IN MONTH
C    LBOT      LAST LINE OF ARRAY TO BE PRINTED
C    LP        ARRAY OF LOGICAL PRINT INDICATORS
C    MTRC      NUMBER OF CANAL MONITORING POINTS
C    NMTR        "       STAGE     "         "
C    MTRC      SUBSCRIPT VALUES OF CANAL MONITOR POINTS
C    MTRP          "        "      STAGE    "      "
C    NCH       NUMBER OF CANALS IN MODEL
C    OLD       ARRAY USED FOR MISCELLANEOUS LISTINGS
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE printlp (imth, idd, idymax, finstor, getfdd,
     & rmonthly_loss_bmp, rmetvol_res, finstor_res, lok_split_option,
     & stagelo_res, rmthly_rf_lok, n_cells, total_small_afact, pond_end,
     & total_cell_stor,ltotal,seepmon,asr_t_cell_volm)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'budg.inc'
```

```
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'et.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'daydump.inc'

      CHARACTER  datetag*256, fdate*24

      INTEGER getfdd(150), ierr, n_cells(max_n_gage_loc_outp),
     & no_of_stg_outp(max_n_gage_loc_outp)

      REAL pond_end(max_ncells), total_small_afact(max_ncells),
     & values(max_ncells),total_cell_stor(max_ncells),
     & seepmon(max_n_lvsp_loc),asr_t_cell_volm(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  MAKE A DATE TAG FOR BINARY FILES

      IF ( lp(22) ) THEN
         CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1,datetag)
      END IF

C  START PRINTING

      IF ( LP(3) ) THEN
         DO 10 I = MAXY,1,-1
            LL = (MINX(I) - 1) *  5 + 1
            WRITE (90,160) 1,B(1:LL),(H(NODE)+POND(NODE), NODE = ISUM(I)
     $      + 1,ISUM(I + 1))
   10    CONTINUE
      END IF
      IF (MONTH .EQ. 12 .AND. IDD .EQ. 31 .AND. LP(4)) THEN
         DO 30 I = MAXY,1,-1
            LL = (MINX(I) - 1) *  5 + 1
            WRITE (90,150) 3,B(1:LL),(FLOAT(IDFL(NODE)) / 30.42, NODE =
     $      ISUM(I) + 1,ISUM(I + 1))
            DO 20 J = ISUM(I) + 1,ISUM(I + 1)
               IDFL(J) = 0
   20       CONTINUE
   30    CONTINUE
      END IF
C-----------------
C  WRITE STAGE, ET, AND PONDING FILES TO DISC
C  IF lp(22) IS .TRUE. OUTPUT IS TO BINARY GRID FILES.  EACH DATA
C  TYPE (PONDING, STAGE, ET) IS OUTPUT TO A DIFFERENT FILE.  THESE
C  FILES SHOULD BE OPENED, AND HEADERS PLACED IN THEM, IN THE
C  INDATA ROUTINE.
C-----------------
C     STAGE
C-----------------
      LBOT = 60
```

```
      IF ( LP(16) ) THEN
         IF ( LP (22) ) THEN
            if (.NOT. lp(25)) THEN
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = H(NODE)+POND(NODE)
               END DO
               CALL GWRITE(40,datetag, VALUES, IERR)
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = STAGEND(NODE)
               ENDDO
               CALL GWRITE(140,datetag,VALUES, IERR)
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = total_cell_stor(NODE)
               ENDDO
               CALL GWRITE(142,datetag,VALUES, IERR)
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = asr_t_cell_volm(NODE)
               ENDDO
               CALL GWRITE(143,datetag,VALUES, IERR)
            ENDIF
         ELSE
            if (.NOT. lp(25)) THEN
               WRITE (70,40) 1,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),1
            ELSE IF( day1 ) THEN
               WRITE (70,50) 1,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),1
            ENDIF
            DO 60 I = LBOT,1,-1
               NPTS = 15 - MINX(I)
               IF (NPTS .LT. 0) NPTS = 0
               LL = (MINX(I) + NPTS - 15) * 5 + 1
               WRITE (70,150) 1,B(1:LL),(STAGEND(NODE), NODE =
     $         ISUM(I) +NPTS +  1,ISUM(I + 1))
C     LL = (MINX(I) - 1) * 5 + 1
C     WRITE (70,1) 4,B(1:LL),(H(NODE) + POND(NODE)
C     *,      NODE = ISUM(I) + 1 ,ISUM(I) + 14)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
   60       CONTINUE
         END IF
      END IF
C-----------------
C     SURFACE PONDING
C-----------------
      IF ( LP(17) ) THEN
         IF ( lp(22) ) THEN
            if (.NOT. lp(25)) THEN
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = POND_END(NODE)
               END DO
               CALL GWRITE(41, datetag, VALUES, IERR)
            ENDIF
         ELSE
            if (.NOT. lp(25)) THEN
               WRITE (70,70) 2,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),2
               WRITE (70,70) 5,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),5
            ELSE IF( day1 ) THEN
               WRITE (70,70) 2,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),2
               WRITE (70,70) 5,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),5
```

```
            ENDIF
            DO 80 I = LBOT,1,-1
               NPTS = 15 - MINX(I)
               IF (NPTS .LT. 0) NPTS = 0
               LL = (MINX(I) + NPTS - 15) * 5 + 1
               WRITE (70,160) 2,B(1:LL),( POND_END(NODE),NODE=ISUM(I) +
     $         NPTS +  1,ISUM(I + 1))
               LL = (MINX(I) - 1) * 5 + 1
               WRITE (70,160) 5,B(1:LL),( POND_END(NODE),NODE=ISUM(I) +
     $          1,ISUM(I) + 14)
   80       CONTINUE
         END IF
      END IF
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF ( LP(18) ) THEN
         IF ( lp(22) ) THEN
            if (idd .EQ. idymax) THEN
c et_total.bin
               if (imth .EQ. ibm) THEN
                  DO NODE = 1, ISUM(MAXY + 1)
                     VALUES(NODE) = 0.0
                  END DO
                  CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
                  CALL GWRITE(42, datetag, VALUES, IERR)
                  CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
               ENDIF
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = ETM(NODE) * 12.
               END DO
               CALL GWRITE(42,datetag, VALUES, IERR)
c rainfall.bin
               if (imth .EQ. ibm) THEN
                  DO NODE = 1, ISUM(MAXY + 1)
                     VALUES(NODE) = 0.0
                  END DO
                  CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
                  CALL GWRITE(44, datetag, VALUES, IERR)
                  CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
               ENDIF
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = RFM(NODE)
               END DO
               CALL GWRITE(44,datetag, VALUES, IERR)
c pumpage.bin
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = WELLS(NODE)
               END DO
               if (imth .EQ. ibm) THEN
                  CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
                  CALL GWRITE(45, datetag, VALUES, IERR)
                  CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
               ENDIF
               CALL GWRITE(45,datetag, VALUES, IERR)
            ENDIF
            IF (.NOT. lp(25)) THEN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
c
c infiltration monthly totals are in inches
               DO NODE = 1, ISUM(MAXY + 1)
clmblmb               VALUES(NODE) = SINFLTM(NODE)
                  VALUES(NODE) = infilt_month(NODE)
               END DO
               IF (use_lec_et .EQ. 0) THEN
clmblmb               CALL GWRITE(48,datetag, VALUES, IERR)
                  CONTINUE
               ELSE
                  CALL GWRITE(46,datetag, VALUES, IERR)
               ENDIF

c percolation monthly totals are in inches
               IF (use_lec_et .NE. 0) THEN
                  DO NODE = 1, ISUM(MAXY + 1)
                     VALUES(NODE) = perc_month(NODE)
                  END DO
                  CALL GWRITE(46,datetag, VALUES, IERR)
               ENDIF
c should solmx() be multiplied by 12. ?
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = SOLMX(NODE)*(1.0
     $             - total_small_afact(NODE)-carfact(NODE))
               END DO
               CALL GWRITE(50,datetag, VALUES, IERR)
c
            ELSE IF(iddump) THEN
c DO gwrites for infilt, perc & solmx
c infiltration dailies are gwritten in inches
               DO node = 1, isum(maxy + 1)
                  values(node) = infilt(node) * 12.
               END DO
               IF (use_lec_et .EQ. 0) THEN
                  CONTINUE
               ELSE
                  call gwrite(46, datetag, values, ierr)
               ENDIF

c percolation dailies are gwritten in inches
               IF (use_lec_et .NE. 0) THEN
                  DO node = 1, isum(maxy + 1)
                     values(node) = perc(node) * 12.
                  END DO
                  call gwrite(46, datetag, values, ierr)
               ENDIF
c solmx() is in feet
               DO node = 1, isum(maxy + 1)
                  values(node) = solmx(node)*(1.0
     $            - total_small_afact(node)-carfact(node))
               END DO
               call gwrite(50, datetag, values, ierr)
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         ELSE
            IF (.NOT. lp(25)) THEN
               WRITE (70,90) 3,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),3
```

```
               WRITE (70,90) 6,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),6
            ELSE IF( day1 ) THEN
               WRITE (70,100) 3,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),3
               WRITE (70,100) 6,MONTH,IDD,IYEAR + IFYR - 1,FDATE(),6
            ENDIF
            if ( .NOT. lp(25) ) THEN
               DO 110 I = LBOT,1,-1
                  NPTS =  15- MINX(I)
                  IF (NPTS .LT. 0) NPTS = 0
                  LL = (MINX(I) + NPTS - 15) * 5 + 1
                  WRITE (70,160) 3,B(1:LL),( 12* ETM(NODE) , NODE = ISUM
     $            (I) + NPTS +  1,ISUM(I + 1))
                  LL = (MINX(I) - 1) * 5 + 1
                  WRITE (70,160) 6,B(1:LL),( 12* ETM(NODE) , NODE = ISUM
     $            (I) +  1,ISUM(I) + 14)
  110          CONTINUE
            ELSE
               CONTINUE
c still have to write daily et computed in agarea.F & ovlnf.F to tape70
            ENDIF
         END IF
      END IF
C-----------------
C WRITE GROUND WATER FLOWS
C-----------------
      IF (LP(20)) THEN
         IF ( lp(22) ) THEN
            if (.NOT. lp(25)) THEN
               if (idd .EQ. idymax) THEN
                  if (imth .EQ. ibm) THEN
c gwrite initial x- & y- snapshots
                     DO NODE = 1, ISUM(MAXY + 1)
                        VALUES(NODE) = 0.0
                     END DO
                     CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
                     CALL GWRITE(49, datetag, VALUES, IERR)
                     CALL GWRITE(49, datetag, VALUES, IERR)
                     CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
                  ENDIF
                  DO IT = 1, 2
                     DO I = 1, ISUM(MAXY + 1)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C IT = 1, FLOW IN THE X DIRECTION
C IT = 2, FLOW IN THE Y DIRECTION
C NOTE: Y FLOW IS COMPUTED IS POSITIVE TO SOUTH,
C       SHOULD BE OUTPUT POSITIVE TO NORTH
c
                        IF (IT .EQ. 1) THEN
                           VALUES(I) = HMVOL(I, IT)/ 43560000.0
                        ELSE
                           VALUES(I) = - HMVOL(I, IT) / 43560000.0
                        ENDIF
                     END DO
                     CALL GWRITE(49, datetag, VALUES, IERR)
                  END DO
               ENDIF
```

```
            ELSE IF (iddump) THEN
c
c         output daily values of groundwater flow
c
               call DSTRING(MONTH,IDAY,IYEAR+IFYR-1,datetag)
C
C           IT = 1, FLOW IN THE X DIRECTION
C
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = HMVOL_daily(NODE, 1) / 43560.
               END DO
               CALL GWRITE(49,datetag, VALUES, IERR)
C
C           IT = 2, FLOW IN THE Y DIRECTION
C           NOTE Y FLOW IS COMPUTED IS POSITIVE TO SOUTH, SHOULD
C           BE POSITIVE TO NORTH
C
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = - HMVOL_daily(NODE, 2) / 43560.
               END DO
               CALL GWRITE(49,datetag, VALUES, IERR)
            ENDIF
         END IF
      END IF
C-----------------
C WRITE SURFACE FLOWS
C-----------------
      IF (LP(20)) THEN
         IF ( lp(22) ) THEN
            if (.NOT. lp(25) .AND. .NOT. lp(19)) THEN
               if (idd .EQ. idymax) THEN
                  if (imth .EQ. ibm) THEN
c gwrite initial x- & y- snapshots
                     DO NODE = 1, ISUM(MAXY + 1)
                        VALUES(NODE) = 0.0
                     END DO
                     CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
                     CALL GWRITE(43, datetag, VALUES, IERR)
                     CALL GWRITE(43, datetag, VALUES, IERR)
                     CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C           IT = 1, FLOW IN THE X DIRECTION
C
                  DO NODE = 1, ISUM(MAXY + 1)
                     VALUES(NODE) = OVLFLO(NODE, 1) / 43560000.
                  END DO
                  CALL GWRITE(43,datetag, VALUES, IERR)
C
C           IT = 2, FLOW IN THE Y DIRECTION
C           NOTE Y FLOW IS COMPUTED IS POSITIVE TO SOUTH, SHOULD
C           BE POSITIVE TO NORTH
C
                  DO NODE = 1, ISUM(MAXY + 1)
                     VALUES(NODE) = - OVLFLO(NODE, 2) / 43560000.
                  END DO
```

```
                  CALL GWRITE(43,datetag, VALUES, IERR)
               ENDIF
            ELSE IF (iddump .AND. .NOT. lp(19)) THEN
c
c         output daily values of surface water flow
c
               call DSTRING(MONTH,IDAY,IYEAR+IFYR-1,datetag)
C
C           IT = 1, FLOW IN THE X DIRECTION
C
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = OVLFLO_daily(NODE, 1) / 43560.
               END DO
               CALL GWRITE(43,datetag, VALUES, IERR)
C
C           IT = 2, FLOW IN THE Y DIRECTION
C           NOTE Y FLOW IS COMPUTED IS POSITIVE TO SOUTH, SHOULD
C           BE POSITIVE TO NORTH
C
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = - OVLFLO_daily(NODE, 2) / 43560.
               END DO
               CALL GWRITE(43,datetag, VALUES, IERR)
            ENDIF
         ELSE
            if (.NOT. lp(25)) THEN
               DO 130 IT = 1,2
                  DO 120 I = MAXY,1,-1
                     LL = (MINX(I) - 1) *  8+ 1
                     WRITE(92,200) IT,B(1:LL),(OVLFLO(NODE,IT) /
     $               4356000., NODE = ISUM(I) + 1, ISUM(I + 1))
  120             CONTINUE
  130          CONTINUE
            ENDIF
         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C    monthly total nodal surface water flow to canal (in acre-ft)
c
c ovflw_to_cnl.bin
      IF (lp(22) .AND. LP(24)) THEN
         if (idd .EQ. idymax) THEN
            if (imth .EQ. ibm) THEN
               DO NODE = 1, ISUM(MAXY + 1)
                  VALUES(NODE) = 0.0
               END DO
               CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
               CALL GWRITE(39, datetag, VALUES, IERR)
               CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
            ENDIF
            DO NODE = 1, ISUM(MAXY + 1)
               VALUES(NODE) = rmonthly_ovlflo_to_canal(NODE)
            END DO
            CALL GWRITE(39,datetag, VALUES, IERR)
         ENDIF
      ENDIF
```

```
C
C    monthly total nodal gw seepage flow to canal (in acre-ft)
c
c seep_to_cnl.bin
      IF (lp(22) .AND. LP(24)) THEN
         if (idd .EQ. idymax) THEN
            if (imth .EQ. ibm) THEN
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = 0.0
               END DO
               CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
               CALL GWRITE(35, datetag, VALUES, IERR)
               CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
            ENDIF
            DO NODE = 1, ISUM(MAXY + 1)
               VALUES(NODE) = rmonthly_seep_to_canal(NODE)
            END DO
            CALL GWRITE(35,datetag, VALUES, IERR)
         ENDIF
      ENDIF
C
C    monthly total canal evaporation (in 1000 ft^3)
c
c canal_evap.bin
      IF (lp(22) .AND. LP(24)) THEN
         if (idd .EQ. idymax) THEN
            if (imth .EQ. ibm) THEN
               DO NODE = 1, ISUM(MAXY+1)
                  VALUES(NODE) = 0.0
               END DO
               CALL DSTRING(MONTH, 1, IYEAR + IFYR - 1, datetag)
               CALL GWRITE(38, datetag, VALUES, IERR)
               CALL DSTRING(MONTH, IDD, IYEAR + IFYR - 1, datetag)
            ENDIF
            DO NODE = 1, ISUM(MAXY + 1)
               VALUES(NODE) = rmonthly_canal_evap(NODE)
            END DO
            CALL GWRITE(38,datetag, VALUES, IERR)
         ENDIF
      ENDIF
C----------------
C     LAKE RAINFALL,ET,AND END OF MONTH STORAGE
C----------------
      IF (runmode .eq. 'SIMUL') THEN
         IF (LP(16)) THEN
            if (lok_split_option .EQ. 0) THEN
               WRITE(80,'(I4,2I3,F6.2,5F10.0)') IFYR+IYEAR-1,MONTH,IDAY
     $         - 1,rmthly_rf_lok,RMETVOL,FINSTOR,rmonthly_loss_bmp
            ELSE
               WRITE(80,'(I4,2I3,F6.2,5F10.0)') IFYR+IYEAR-1,MONTH,IDAY
     $         - 1,rmthly_rf_lok,RMETVOL,FINSTOR,rmonthly_loss_bmp,
     $         rmetvol_res ,finstor_res
            ENDIF
         ENDIF
      ELSE
         IF (LP(16)) THEN
            if (lok_split_option .eq. 0) then
```

```
              WRITE(80,'(I4,2I3,F6.2,5F10.0)') IFYR+IYEAR-1,MONTH,IDAY
     *        - 1,LOKMORN(MONTH),RMETVOL,FINSTOR,rmonthly_loss_bmp
            else
              WRITE(80,'(I4,2I3,F6.2,5F10.0)') IFYR+IYEAR-1,MONTH,IDAY
     *        - 1,LOKMORN(MONTH),RMETVOL,FINSTOR,rmonthly_loss_bmp
     *        ,rmetvol_res,finstor_res
            endif
         ENDIF
      ENDIF
c
c      MONTHLY TOTAL LEVEE SEEPAGE (ACRE-FT)
c
      IF (.NOT. LP(25) .AND. .NOT. DAY1) THEN
         WRITE(32,175)IFYR+IYEAR-1,MONTH,(SEEPMON(L),L=1,LTOTAL)
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C----------------
C  THE OLD ARRAY IS USED FOR MISCELLANEOUS ARRAY LISTINGS
C       RESET IT TO ZERO AFTER PRINTING.
C-----------------
C     DO 30 I = LBOT,1,-1
C       DO 30 L = ISUM(I) + 1,ISUM(I + 1)
C       ETM(L) = 0.0
C 30    CONTINUE
C-----------------
C  WRITE CA-3 MONITOR POINTS FOR EASY REFERENCE
C-----------------
      IF ( NMTR .GT. 0 ) THEN
         DO I = 1,NMTR
            no_of_stg_outp(i) = 1
            DO k =1,n_cells(i)
               IF (noresincell(MTRP(i,k)) .GT. 0) THEN
                  no_of_stg_outp(i) = noresincell(MTRP(i,k))
               ENDIF
            ENDDO
         ENDDO
         IF (IDD .NE. IDYMAX .AND. .NOT. DAY1) RETURN
         IF (lok_split_option .EQ. 0) THEN
            PRINT 180, MONTH,IDD,IFYR+IYEAR-1,(NINT(BAGP(I)/
     $      43560000.),I=1,NEAABSN),
     $     ((rmean_ACSTG(I,IR),IR=1,no_of_stg_outp(i)),I = 1,NMTR)
     $     ,STAGELO
         ELSE
            PRINT 180, MONTH,IDD,IFYR+IYEAR-1,(NINT(BAGP(I)/
     $      43560000.),I=1,NEAABSN),
     $     ((rmean_ACSTG(I,IR),IR=1,no_of_stg_outp(i)),I = 1,NMTR)
     $     ,STAGELO,STAGELO_RES
         ENDIF
         DO 140 I= 1,NEAABSN
  140    BAGP(I) = 0.0
      END IF
C
C-----------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      RETURN
   40 FORMAT (I1,1H1,T40,'MONTH END STAGE',I6,'/',I4,'/', I4,T120, A24,
```

```
   $/I1)
 50 FORMAT (I1,1H1,T40,'END-OF-DAY STG ',I6,'/',I4,'/', I4,T120, A24,
   $/I1)
 70 FORMAT (I1,1H1,T40,'  SURFACE PONDING ',I4,'/', I4,'/',I4,T120,
   $A24,/,I1)
 90 FORMAT (I1,1H1,T40,'MONTHLY ET (IN.)',I4,'/',I2,'/', I4,T120,A24,
   $/,I1)
100 FORMAT (I1,1H1,T40,'DAILY ET   (IN.)',I4,'/',I2,'/', I4,T120,A24,
   $/,I1)
150 FORMAT (I1,A,42F5.1)
160 FORMAT (I1,A,42F5.1)
170 FORMAT (1X,42F5.2)
175 FORMAT(3X,I5,1X,I2,100F6.0)
180 FORMAT (1X,I2,'/',I2,'/',I4,T13,4I6,T40,75(F6.2,1X))
190 FORMAT (2I3,I5,33F6.2)
200 FORMAT(I1,A,42F8.2)
210 FORMAT(I1,A,42I7)
    END
```

qmax_one_reach.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: qmax_one_reach.F,v 1.7 2003/07/03 20:07:40 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/qmax_one_reach.F,v $
C   $Name: rel-5-5 $
C
C
C  This SUBROUTINE calculates the conveyance capacity for
C  a spillway-canal-spillway reach within the eaa.
C  input variables:
C    suhw    = headwater for the upstream spillway
C    sltw    = tailwater for the downtream spillway
C    istrup  = upstream spillway number
C    istrdn  = downtream spillway number
C    ireach  = reach number for the canal
C    iagbsn  = basin number where system is located
C
C  output variables:
C    qconvey = system conveyance capacity
C    struq   = upstream spillway discharge
C    qcanal  = canal discharge
C    strlq   = downstream spillway discharge
C    chw     = canal headwater
C    ctw     = canal tailwater
C
C  List of Reaches
C  Base Case
C  1. EAA Miami Canal:           S354HW to S8TW
C  2. EAA North New River Canal: S351HW to S7TW
C  3. EAA Hillsboro Canal:       S351TW to S6HW
C  4. EAA West Palm Beach Canal: S352TW to S5AHW
C  Future Base
C  5. EAA Miami Canal:           S354TW to S8newHW
C  6. EAA North New River Canal: S351TW to S7newHW
C  7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
C  8. EAA Miami Canal:           S8newTW to S8HW
```

```
C  9. EAA North New River Canal: S7newTW to S7HW
C
C  List of Spillways
C    eaa spillway name
C  1.   s8
C  2.   s7
C  3.   s8new
C  4.   s7new
C  5.   s354             stagelo to lo_str_tw(imiami)
C  6.   s351             stagelo to lo_str_tw(innr)
C  7.   s352             stagelo to lo_str_tw(iwpb)
C  8.   s354 forward pumping
C  9.   s351 forward pumping
C  10.  s352 forward pumping
C
C  Basin numbers
C  1.   imiami
C  2.   innr
C  3.   iwpb
C  4.   ihill
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE qmax_one_reach (suhw, sltw, istrup, istrdn, ireach,
     & iagbsn, qconvey, struq, qcanal, strlq, chw, ctw, ifacteaa)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'eaacap.inc'
      INCLUDE 'mgateop.inc'
      INCLUDE 'fin_go.inc'

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  tolerq = convergence criterion for matching flows
      tolerq = 1.

C  itermax = maximum allowable number of iterations
      itermax = 100

C  for gravity flow
      IF (.not. LFORPUMP(ireach) .or. (ireach.ne.1 .and.
     $     ireach.ne.2)) then

C  normal operations
         IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
            IF (suhw .GT. rmaxstg(ireach)) THEN
               chw = rmaxstg(ireach)
            ELSE
               chw = suhw
               struq = 0.0
            ENDIF
```

```
         ELSE
            chw = suhw
            struq = 0.0
         ENDIF
c
         IF (suhw.LE.sltw) THEN
            qconvey = 0.0
            struq = 0.0
            qcanal = 0.0
            strlq = 0.0
c
            ctw = chw
            GO TO 20
         ENDIF
         IF (chw .GT. sltw) THEN
            ctw = sltw
         ELSE
            qconvey = 0.0
            struq = 0.0
            IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
               IF (suhw .GT. chw) THEN
                  struq = -1.0
                  call eaaspill(struq,suhw,chw,istrup,ifacteaa)
               ENDIF
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            qcanal = 0.0
            strlq = 0.0
c
            ctw = chw
            GO TO 20
         ENDIF
         istype = 1
         istr = istrdn
         delta0 = (chw-ctw) / 25.
         delta1 = delta0
         IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
            struq = -1.0
            call eaaspill(struq,suhw,chw,istrup,ifacteaa)
         ENDIF
c routine ds_flow(istype,istr,ireach,iagbsn,chw,
c                 ctw,stw,delta0,toler,canalq,qlstrr,shw,ifacteaa)
         call ds_flow(istype,istr,ireach,iagbsn,chw,ctw,sltw, delta0,
     $   tolerq,itermax,qcanal,strlq,shw,ifacteaa)
         IF (strlq .LE. struq) THEN
            qconvey = strlq
c       OR  qconvey = qcanal
c Actually, after ds_flow is called,
c           qcanal == strlq (=qlstrr)
            GO TO 20
         ENDIF
c
         iter = 0
   10    ctw = ctw - delta1
         strlq = -1.0
         call eaaspill(strlq,ctw,sltw,istrdn,ifacteaa)
         qcanal = strlq
```

```
         chw = -1.0
         call eaa_prof_lu_table(qcanal,chw,ctw,ireach,ifacteaa)
         IF (chw .GE. suhw) GO TO 10
         struq = -1.0
         call eaaspill(struq,suhw,chw,istrup,ifacteaa)
c
         iter = iter + 1
         IF (iter.EQ.itermax)GO TO 20
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (strlq-struq .GT. tolerq) THEN
c continue to decrease ctw such that struq increases and strlq decreases
            GO TO 10
         ENDIF
         IF (struq-strlq .GT. tolerq) THEN
c overshot estimation of new ctw; redo with 1/2 delta1
            ctw = ctw + delta1
            delta1 = delta1 / 2.
            GO TO 10
         ENDIF
      ELSE
c low LOK stages - require forward pumping
c
         struq = -1.0
         chw = amin1(rmaxstg(ireach),stagelo+rmax_lift(ireach))
         call eaaspill (struq,stagelo,chw,istrup,ifacteaa)
         strlq = forw_pump_cap(ireach)
         ctw = -1.0
         call eaaspill(strlq,ctw,sltw,istrdn,ifacteaa)
         chw = -1.0
         qcanal = forw_pump_cap(ireach)
         rmax_cuhw = amin1(rmaxstg(ireach),stagelo
     $   +rmax_lift(ireach))
         call eaa_prof_lu_table(qcanal,chw,ctw,ireach,ifacteaa)
         IF (chw .gt. rmax_cuhw) then
            chw = rmax_cuhw
            istype = 1
            istr = istrdn
            delta0 = (chw-ctw) / 25.
            delta1 = delta0
c routine ds_flow(istype,istr,ireach,iagbsn,chw,
c                 ctw,stw,delta0,toler,canalq,qlstrr,shw,ifacteaa)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            call ds_flow(istype,istr,ireach,iagbsn,chw,ctw,sltw, delta0,
     $      tolerq,itermax,qcanal,strlq,shw,ifacteaa)
c
         ENDIF
      ENDIF
c adjustment within tolerance; exit
   20 qconvey = amin1(struq,qcanal,strlq)
c         print *,'qconvey: ',qconvey
c         write(*,'(12f10.3)')struq,qcanal,strlq,suhw,chw,ctw,sltw
      RETURN
      END
```

qmax_two_reaches.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: qmax_two_reaches.F,v 1.8 2003/07/03 20:07:41 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/qmax_two_reaches.F,v $
C   $Name: rel-5-5 $
C
C
C  This SUBROUTINE calculates the conveyance capacity for
C  a spillway-canal-spillway reach within the eaa.
C  input variables:
C    suhw    = headwater for the upstream spillway
C    sltw    = tailwater for the downtream spillway
C    istrup  = upstream spillway number
C    istrdn  = downtream spillway number
C    ireach  = reach number for the canal
C    iagbsn  = basin number where system is located
C
C  output variables:
C    qconvey = system conveyance capacity
C    struq   = upstream spillway discharge
C    qcanal  = canal discharge
C    strlq   = downstream spillway discharge
C    chw     = canal headwater
C    ctw     = canal tailwater
C
C  List of Reaches
C  Base Case
C  1. EAA Miami Canal:           S354HW to S8TW
C  2. EAA North New River Canal: S351HW to S7TW
C  3. EAA Hillsboro Canal:       S351TW to S6HW
C  4. EAA West Palm Beach Canal: S352TW to S5AHW
C  Future Base
C  5. EAA Miami Canal:           S354TW to S8newHW
C  6. EAA North New River Canal: S351TW to S7newHW
C  7. EAA Hillsboro Canal:       S351TW to S6HW w/ new x-sections
C  8. EAA Miami Canal:           S8newTW to S8HW
```

```
C  9. EAA North New River Canal: S7newTW to S7HW
C
C  List of Spillways
C    eaa spillway name
C  1.   s8
C  2.   s7
C  3.   s8new
C  4.   s7new
C  5.   s354             stagelo to lo_str_tw(imiami)
C  6.   s351             stagelo to lo_str_tw(innr)
C  7.   s352             stagelo to lo_str_tw(iwpb)
C
C  Basin numbers
C  1.   imiami
C  2.   innr
C  3.   iwpb
C  4.   ihill
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE qmax_two_reaches (suhw, sltw, istrup, istrdn, ireach,
     & iagbsn, qconvey, struq, qcanal1, strlq1, chw, ctw, ifacteaa)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'eaacap.inc'
      INCLUDE 'mgateop.inc'
      INCLUDE 'fin_go.inc'

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C tolerq = convergence criterion for matching flows
      tolerq = 1.

C itermax = maximum allowable number of iterations
      itermax = 100

C for gravity flow
      IF (.not. LFORPUMP(ireach) .or. (ireach.ne.1 .and.
     $     ireach.ne.2)) then

C normal operations
         IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
            IF (suhw .GT. rmaxstg(ireach)) THEN
               chw = rmaxstg(ireach)
            ELSE
               chw = suhw
               struq = 0.0
            ENDIF
         ELSE
            chw = suhw
            struq = 0.0
```

```
         ENDIF
c
         IF (suhw.LE.sltw) THEN
            qconvey = 0.0
            struq = 0.0
            qcanal1 = 0.0
            qcanal2 = 0.0
            strlq1 = 0.0
c
            ctw = chw
            GO TO 30
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (chw .GT. sltw) THEN
            ctw = sltw
         ELSE
            IF (chw .LE. pmp_int(is6bsn-6)) THEN
               qconvey = 0.0
               struq = 0.0
               IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
                  IF (suhw .GT. chw) THEN
                     struq = -1.0
                     call eaaspill(struq,suhw,chw,istrup,ifacteaa)
                  ENDIF
               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               qcanal  = 0.0
               qconvey1 = 0.0
               qconvey2 = 0.0
               qcanal1 = 0.0
               qcanal2 = 0.0
               strlq1  = 0.0
c
               ctw = chw
               GO TO 30
            ELSE
               qconvey1 = 0.0
               qcanal1 = 0.0
               strlq1  = 0.0
               IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
                  struq = -1.0
                  call eaaspill(struq,suhw,chw,istrup,ifacteaa)
               ENDIF
               GO TO 10
            ENDIF
         ENDIF
         istype = 1
         istr = istrdn
         delta0 = (chw-ctw)/25.
         delta1 = delta0
         IF (ireach .EQ. 1 .OR. ireach .EQ. 2) THEN
            struq = -1.0
            call eaaspill(struq,suhw,chw,istrup,ifacteaa)
         ENDIF
c routine ds_flow(istype,istr,ireach,iagbsn,chw,
c                 ctw,stw,delta0,toler,canalq,qlstrr,shw,ifacteaa)
         call ds_flow(istype,istr,ireach,iagbsn,chw,ctw,sltw, delta0,
```

```
     $   tolerq,itermax,qcanal1,strlq1,shw,ifacteaa)
c
         qcanal1 = amin1(qcanal1,strlq1)
   10    qcanal2 = -1.0
         ireach = 3
         ctw2 = pmp_int(is6bsn - 6)
         IF (chw .GT. ctw2) THEN
            call eaa_prof_lu_table(qcanal2,chw,ctw2,ireach,ifacteaa)
         ELSE
            qcanal2 = 0.0
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         qconvey_total = qcanal1 + qcanal2
         IF (struq .GE. qconvey_total) go to 30
c
         iter = 0
   20    chw = chw - delta1
         struq = -1.0
         call eaaspill(struq,suhw,chw,istrup,ifacteaa)
         call ds_flow(istype,istr,ireach,iagbsn,chw,ctw,sltw, delta0,
     $   tolerq,itermax,qcanal1,strlq1,shw,ifacteaa)
         qcanal1 = amin1(qcanal1,strlq1)
         qcanal2 = -1.0
         IF (chw .GT. ctw2) THEN
            call eaa_prof_lu_table(qcanal2,chw,ctw2,3,ifacteaa)
         ELSE
            qcanal2 = 0.0
         ENDIF
         qconvey_total = qcanal1 + qcanal2
c
         iter = iter + 1
         IF (iter .EQ. itermax)GO TO 30
c
         IF (qconvey_total-struq.GT.tolerq) THEN
c continue to decrease ctw such that struq increases and strlq decreases
            GO TO 20
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (struq-qconvey_total.GT.tolerq) THEN
c overshot estimation of new ctw; redo with 1/2 delta1
            chw = chw + delta1
            delta1 = delta1/2.
            GO TO 20
         ENDIF
      ELSE
c  forward pumping when LOK is low
         chw = suhw
         istype = 1
         istr = istrdn
         ctw = sltw
         delta0 = (chw-ctw)/25.
         delta1 = delta0
         ireach = 3
         ctw2 = pmp_int(is6bsn - 6)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         iter = 0
   28    chw = chw - delta1
```

```
         struq = -1.0
         call eaaspill(struq,suhw,chw,istrup,ifacteaa)
c routine ds_flow(istype,istr,ireach,iagbsn,chw,
c                 ctw,stw,delta0,toler,canalq,qlstrr,shw,ifacteaa)
         call ds_flow(istype,istr,ireach,iagbsn,chw,ctw,sltw, delta0,
     $   tolerq,itermax,qcanal1,strlq1,shw,ifacteaa)
         qcanal1 = amin1(qcanal1,strlq1)
         qcanal2 = -1.0
         IF (chw .GT. ctw2) THEN
            call eaa_prof_lu_table(qcanal2,chw,ctw2,ireach,ifacteaa)
         ELSE
            qcanal2 = 0.0
         ENDIF
         qconvey_total = qcanal1 + qcanal2
c
         iter = iter + 1
         IF (iter .EQ. itermax)GO TO 30
c
         IF (qconvey_total-struq.GT.tolerq) THEN
c continue to decrease ctw such that struq increases and strlq decreases
            GO TO 28
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (struq-qconvey_total.GT.tolerq) THEN
c overshot estimation of new ctw; redo with 1/2 delta1
            chw = chw + delta1
            delta1 = delta1/2.
            GO TO 28
         ENDIF
      ENDIF
c adjustment within tolerance; exit
   30 qconvey = amin1(struq,qconvey_total)
c         print *,'qconvey: ',qconvey
c         write(*,'(12f10.3)')struq,qcanal,strlq,suhw,chw,ctw,sltw
      RETURN
      END
```

qpulse.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: qpulse.F,v 1.10 2005/06/10 13:29:16 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/qpulse.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine determines the pulse release from Lake
C   Okeechobee to tidewater called for by operational schedule
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE qpulse (rstgpls, stglake, iplslevel, slpulse, calpulse,
     & month, iday, mzone)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'opersched.inc'

      REAL qplscal(max_n_puls_rel), qplssl(max_n_puls_rel),
     & rstgpls(max_n_puls_rel)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C definition of terms:
C  input:
C   iplsday = number of days before today by which today's
C             stage will be compared against; min = 0,
C             max = 10
C   rstgpls = reference Lake stage which was the Lake's stage
C             iplsday ago
C note: IF iplsday is input, THEN rstgpls is used as an array
C       ELSE, rstgpls is input as a scalar
C   stglake = beginning of day stage for Lake Okeechobee
C   ipulse = flag to make pulse release or not for today;
C            always zero IF Lake is above regulation (i.e.
```

```
C            in either zone A, B or C)
C   iplslevel = pulse release level number
C            : 1, 2 or 3
C  output:
C   slpulse = required pulse release to St. Lucie estuary
C             for today
C   calpulse = required pulse release to Caloosahatchee
C             Esturay for today
C  others:
C   plsstg1 = stage corresponding to level 1, pulse zone
C
C check whether today is within a 10-day pulse release window

      IF (ipulse .EQ. 0) THEN

C determine IF Lake stage is within the 3-level pulse zone
C note: this SUBROUTINE is called every time the stage is
C       below regulation (i.e. below LOK zone C)

         call bpts(iday,month,999.,1,mzone_puls_bot,1,plsstg1)
         IF (stglake .GE. plsstg1)THEN

C determine how was today's stage reached

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            IF (mzone .EQ. 2) THEN
               slope_reg_sched_za = stg_lok_sched(1,iplsday) -
     $         stg_lok_sched(1,1)
               slope_reg_sched = stg_lok_sched(mzone,iplsday) -
     $         stg_lok_sched(mzone,1)
               slope_reg_sched = amax1(slope_reg_sched_za ,
     $         slope_reg_sched)
            ELSE
               slope_reg_sched = stg_lok_sched(mzone,iplsday) -
     $         stg_lok_sched(mzone,1)
            ENDIF
            slope_lok_stg = rstgpls(iplsday) - stglake
            IF (slope_lok_stg .GT. slope_reg_sched .and. iflex_pulse_opt
     $      .EQ. 1) THEN
c today's stage was reached with a declining rate greater than
c regulation schedule; thus
c pulse releases are not allowed
               slpulse  = 0.0
               calpulse = 0.0
               RETURN
            ELSE
c today's stage was reached with a declining rate less than reg
c schedule,THEN pulse releases are allowed
c assign appropriate pulse release level
               DO 10 i = 1, no_of_pulse_rel(iplslevel)
                  qplssl(i)  = qplsl(iplslevel,i)
                  qplscal(i) = qplscl(iplslevel,i)
   10          CONTINUE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c make release
               slpulse  = qplssl(1)
               calpulse = qplscal(1)
```

```
c              print *,slpulse,calpulse,iplslevel,qplsl(iplslevel,1)
c    $            ,qplscl(iplslevel,1),'  in qpulse'
c set flag to one and advance counter to second day of
c 10-day hydrograph of pulse releases
               IF (no_of_pulse_rel(iplslevel) .EQ. 1) THEN
                  ipulse = 0
                  ipday = 1
               ELSE
                  ipulse = 1
                  ipday = 2
               ENDIF
            ENDIF
         ELSE
            RETURN
         ENDIF
      ELSE
c CONTINUE with current level of pulse release
c make release
c        print *,ipday,iplslevel,no_of_pulse_rel
c    $   (iplslevel),qplscal(ipday)
         IF (ipday .GT. 0) THEN
            slpulse = qplssl(ipday)
            calpulse = qplscal(ipday)
         ENDIF
         IF (ipday .EQ. no_of_pulse_rel(iplslevel)) THEN
c last day of pulse release reached; reset flag to zero
            ipulse = 0
         ELSE
c advance counter to next day of 10-day hydropgraph of
c pulse releases
            ipday = ipday + 1
         ENDIF
      ENDIF
      RETURN
      END
```

rain_driven_flow_stage_targ.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: rain_driven_flow_stage_targ.F,v 1.12 2005/06/10 13:29:16 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/rain_driven_flow_stage_targ.F,v
$
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE DETERMINES 1) THE NON-REGULATORY COMPONENT OF TARGET
C   FLOW TO ENP SHARK RIVER SLOUGH BASED ON CURRENT RAINFALL PLAN ,IF
C   TARGET FLOW IN NOT PRE-PROCESSED, AND 2) STATUS OF EVERGLADES
C   SYSTEM RELATIVE TO STAGE TARGETS FOR RAIN DRIVEN OPERATIONS.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE rain_driven_flow_stage_targ (nzone, regstg_wca, wcaenv,
     & ibot_zone_indx, itop_zone_indx, srsdmnd, n_intcnl, jday, nyday,
     & ntotal_no_import_areas, ntotal_no_import_areas_const, iwca3a)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'wrout.inc'

      DIMENSION etsm(75), etsmw(10), etw(62),
     & ibot_zone_indx(max_n_stor_w_oper_sch),
     & itop_zone_indx(max_n_stor_w_oper_sch), n_intcnl(max_n_wcas),
     & nzone(max_n_stor_w_oper_sch), qm(62),
     & regstg_wca(max_n_wcas,max_n_zones_wca), rf(75), rfmw(62),
```

```
     &  rfw(10), srsdmnd(366)

      CHARACTER wcaenv(max_n_wcas)*5

      DATA etw /1.06,1.02,0.97,0.91,0.87,0.82,0.80,0.78,0.76,0.75,0.75
     $,0.77,0.81,0.83,0.88,0.94,1.01,1.08,1.15,1.22,1.29,1.36,1.41
     $,1.47,1.51,1.54,1.58,1.60,1.61,1.62,1.61,1.61,1.59,1.55,1.51
     $,1.49,1.45,1.45,1.43,1.42,1.42,1.42,1.39,1.40,1.36,1.34,1.31
     $,1.27,1.25,1.21,1.16,1.12,1.06,1.02,0.97,0.91,0.87,0.82,0.80
     $,0.78,0.76,0.75/
      DATA qm /1220.,1084.4,936.5,819.8,688.8,566.2,462.3,376.4
     $,320.8,281.3,246.6,216.2,185.8,152.7,129.9,109.8,87.7,69.4,50.4
     $,34.7,24.6,25.5,30.7,31.8,30.5,29.0,24.0,16.4,12.9,11.5,17.3
     $,26.8,39.5,53.9,75.1,93.5,111.3,128.3,148.0,162.0,174.5,185.1
     $,199.2,223.4,256.5,310.3,384.6,540.1,778.9,994.9,1129.6,1229.0
     $,1220.0,1084.4,936.5,819.8,688.8,566.2,462.3,376.4,320.8,281.3/
      DATA rfmw /1.19,0.92,0.56,.40,.32,.29,.27,.24,.24,.24,. 27,.25,.35
     $,.33,.35,.35,.35,.38,.40,.33,.43,.43,.54,.59,.60,.69,.80,.92,1.08
     $,1.21,1.43,1.57,1.67,1.83,1.98,2.10,2.05,1.93,1.95,1.92,1.83,1.72
     $,1.70,1.73,1.89,1.90,2.00,2.16,2.07,1.88,1.61,1.38,1.19,0.92,0.56
     $,.40,.32,.29,.27,.24,.24,.24/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  ENP Rainfall Based Discharge to Shark River Slough.
C  If pre-processed read-in option is true for Rainfall formula amt
C  assign today's read-in value to RF formula variable

      IF (type_flow_across_ttrail .NE. 'MINDEL' .AND. opt_ft_input) THEN
         IF (type_target .EQ. 'TOTAL' .OR. type_flow_across_ttrail .EQ.
     $   'RFPLAN ') FRF = srsdmnd(jday)
         IF (type_target .NE. 'TOTAL' .AND. type_flow_across_ttrail .EQ.
     $      'NSMFLOW') FRF = srsdmnd_w(jday) + srsdmnd_e(jday)
      ENDIF

c IF not true, compute the amt here
C
C
c  Flow Targets for SRS in ENP
c
c
C  INPUT FIRST 10 WEEKS OF RF AND ET FOR RAINFALL FORMULA
C  DONE ONLY IF opt_ft_input = 0
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (type_flow_across_ttrail .NE. 'MINDEL' .AND. .NOT. opt_ft_input
     $ .AND. type_flow_across_ttrail .NE. 'NSMFLOW') THEN
         IF (DAY1 ) THEN
            DO 10 ID = 1,70
               READ(59,'(2F6.0)') RF(ID),ETSM(ID)
   10       CONTINUE
         ELSE
            DO 20 ID = 1,69
               RF(ID) = RF(ID+1)
               ETSM(ID) = ETSM(ID+1)
   20       CONTINUE
            READ(59,'(2F6.0)') RF(70),ETSM(70)
```

```
         ENDIF
C
         DO 30 IFR = 1,10
            RFW(IFR) = 0.0
            ETSMW(IFR) = 0.0
   30    CONTINUE
C
         IWK = 1
         DO 40 ID = 1,70
            RFW(IWK) = RFW(IWK) + RF(ID)
            ETSMW(IWK) = ETSMW(IWK) + ETSM(ID)
            IF (MOD(ID,7) .EQ. 0) IWK = IWK + 1
   40    CONTINUE
         IPWK = (NYDAY - 1) / 7 + 1
         IW = IPWK + 9
         IWS = IW - IPWK + 1
         IF (DAY1) THEN
            QMP = QM(IW)
            IF( LP(1)) THEN
               READ(55,'(2F10.0)') FRFP,FRF
               CLOSE(55)
            ELSE
               FRFP = QMP
            ENDIF
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (MOD(NYDAY,7) .EQ. 1)THEN
            RL1 = RFW(IWS) - RFMW(IW) + RFW(IWS - 1) - RFMW(IW - 1) -
     $      0.8 * (ETSMW(IWS) - ETW(IW) + ETSMW(IWS - 1) - ETW(IW - 1))
C
            RL2 = RFW(IWS - 2) - RFMW(IW - 2) + RFW(IWS - 3) - RFMW
     $      (IW - 3) + RFW(IWS - 4) - RFMW(IW - 4) + RFW(IWS - 5) -
     $      RFMW(IW - 5) - 0.8 * (ETSMW(IWS - 2) - ETW(IW - 2) + ETSMW
     $      (IWS - 3) -ETW(IW - 3) + ETSMW(IWS - 4) - ETW(IW - 4) +
     $          ETSMW(IWS - 5) - ETW(IW-5))
C
            RL3 = RFW(IWS - 6) - RFMW(IW - 6) + RFW(IWS - 7) - RFMW
     $          (IW - 7) + RFW(IWS - 8) - RFMW(IW - 8) + RFW(IWS - 9) -
     $          RFMW(IW - 9) - 0.8 * (ETSMW(IWS - 6) - ETW(IW - 6) + ETSMW
     $      (IWS - 7) - ETW(IW - 7) + ETSMW(IWS - 8) - ETW(IW - 8) +
     $      ETSMW(IWS - 9) - ETW(IW - 9))
C
            IF (RL1 .LT. 0.0) THEN
               CR1 = 52.22
            ELSE
               CR1 = 70.58
            ENDIF
C
            IF (RL2 .LT. 0.0) THEN
               CR2 = 0.0
            ELSE
               CR2 = 9.49
            ENDIF
C
            IF (RL3 .LT. 0.0) THEN
               CR3 = 0.0
            ELSE
```

```
               CR3 = 9.18
            ENDIF
C
            CQ = 0.9069
C
            FRF = QM(IW + 1) + CQ * (FRFP - QMP) + CR1 * RL1 + CR2 * RL2
     $      + CR3 * RL3
            IF (FRF .LT. 0.0) FRF = 0.0
            QMP = QM(IW + 1)
            FRFP = FRF
         ENDIF
      ENDIF
      IF (type_flow_across_ttrail .EQ. 'STAGETG') FRF = 0.0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c-------------------------------------------
c   rain driven stage target setup
c-------------------------------------------
c
c   read in daily import stages for Everglades
c
      IF (runmode .ne. 'CALIB') THEN
        IF (wca_import_variation .EQ. 'DAILY' .OR. opt_flow_to_ts .EQ.
     $  'TSRFPLN' .OR. opt_st_input) THEN
          read (83,'(11x,45f6.0)') (daily_stg_import_wca(i),i= 1,
     $    ngrid_cells_daily_total)
        ENDIF
      ENDIF
c
c
      DO isa = 1,ntotal_no_import_areas+1
         LWSLO(ISA) = .FALSE.
         LRGLO(ISA) = .FALSE.
         LWSLO_eaares(ISA) = .FALSE.
      END DO
c
c set up array for for downstream nodal stage limit to struc flow
c  in LEC service areas
c
      DO iwca = 1,nwcas
c
c  determine regulation sched for wcas on daily basis
c
         DO izone = 1,nzone(iwca+1)
            iwca_zone = izone
            CALL bpts(iday,month,10.0,iwca + 1,iwca_zone,1,rgstg)
            regstg_wca(iwca,izone) = rgstg
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (wca_import_variation .EQ. 'DAILY'.OR. opt_flow_to_ts .EQ.
     $'TSRFPLN' .OR. opt_st_input) THEN
         DO isa = 1,ntotal_no_import_areas
            IF (ngrid_cells_daily(isa) .GT. 0) THEN
               stage_import_wca(isa) = 0.0
               avg_sim_wca_stg(isa) = 0.0
               DO i = 1,ngrid_cells_daily(isa)
```

```
                  stage_import_wca(isa) = stage_import_wca(isa) +
     $            daily_stg_import_wca(ngage_index_env(isa,i,1)) /
     $            ngrid_cells_daily(isa)
c
c  compute simulated stages
c
                  depthgw = (ells(node_import_daily(isa,i)) - h
     $            (node_import_daily(isa,i))) * s(node_import_daily
     $            (isa,i))
                  IF (pond(node_import_daily(isa,i)) .GT. depthgw) THEN
                     wca_nodal_stage = pond(node_import_daily(isa,i)) -
     $               depthgw + ells(node_import_daily(isa,i))
                  ELSE
                     wca_nodal_stage = (pond(node_import_daily(isa,i)) -
     $               depthgw)/s(node_import_daily(isa,i)) + ells
     $               (node_import_daily(isa,i))
                  ENDIF
                  avg_sim_wca_stg(isa) = avg_sim_wca_stg(isa) +
     $            wca_nodal_stage / ngrid_cells_daily(isa)
               END DO
               LFLWTRG(ISA) = avg_sim_wca_stg(isa) .LT. stage_import_wca
     $         (isa)-offset(isa,1)
            ENDIF
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      ELSE
         IF (wca_import_variation .EQ. 'CONST') THEN
            DO isa = 1,ntotal_no_import_areas_const
               IF (nmpts_wca(isa) .GT. 0) THEN
                  stage_import_wca(isa) = avg_import_stage(isa)
                  avg_sim_wca_stg(isa) = 0.0
                  DO i = 1,nmpts_wca(isa)
                     depthgw = (ells(node_import(isa,i)) - h(node_import
     $               (isa,i))) * s(node_import(isa,i))
                     IF (pond(node_import(isa,i)) .GT. depthgw) THEN
                        wca_nodal_stage = pond(node_import(isa,i)) -
     $                  depthgw + ells(node_import(isa,i))
                     ELSE
                        wca_nodal_stage = (pond(node_import(isa,i)) -
     $                  depthgw)/s(node_import(isa,i)) + ells
     $                  (node_import(isa,i))
                     ENDIF
                     avg_sim_wca_stg(isa) = avg_sim_wca_stg(isa) +
     $               wca_nodal_stage / nmpts_wca(isa)
                  END DO
                  LFLWTRG(ISA) = avg_sim_wca_stg(isa) .LE.
     $            stage_import_wca(isa) - offset(isa,1)
               ENDIF
            END DO
         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C  COMPUTE IMPORT STAGES FOR RAIN DRIVEN OPS IN EVERGLADES/WETLANDS
C
      IF (wca_import_variation .EQ. 'DAILY'.OR. opt_flow_to_ts .EQ.
     $'TSRFPLN' .OR. opt_st_input) THEN
```

```
c
c  use time-varying (daily) targets
c
         DO isa = 1,ntotal_no_import_areas
            IF ((isa .LE. nwcas .AND. wcaenv(isa) .EQ. 'TRUE ') .OR.
     $      isa .GT. nwcas) THEN
c
c  rain-driven ops
c
               IF (ngrid_cells_daily(isa) .GT. 0) THEN
                  stage_import_wca(isa) = 0.0
                  avg_sim_wca_stg(isa) = 0.0
                  DO i = 1,ngrid_cells_daily(isa)
                     stage_import_wca(isa) = stage_import_wca(isa) +
     $               daily_stg_import_wca(ngage_index_env(isa,i,1)) /
     $               ngrid_cells_daily(isa)
c
c  compute simulated stages
c
                     depthgw = (ells(node_import_daily(isa,i)) - h
     $               (node_import_daily(isa,i))) * s(node_import_daily
     $               (isa,i))
                     IF (pond(node_import_daily(isa,i)) .GT. depthgw)
     $               THEN
                        wca_nodal_stage = pond(node_import_daily(isa,i))
     $                  - depthgw + ells(node_import_daily(isa,i))
                     ELSE
                        wca_nodal_stage = (pond(node_import_daily(isa,i)
     $                  ) - depthgw)/s(node_import_daily(isa,i)) + ells
     $                  (node_import_daily(isa,i))
                     ENDIF
                     avg_sim_wca_stg(isa) = avg_sim_wca_stg(isa) +
     $               wca_nodal_stage / ngrid_cells_daily(isa)
                  END DO
                  LWSLO(ISA) = avg_sim_wca_stg(isa).LT.stage_import_wca
     $            (isa) - offset(isa,1)
                  LRGLO(ISA) = avg_sim_wca_stg(isa) .LE.
     $            stage_import_wca(isa) + abs(offset(isa,2)) + 0.25
                  LWSLO_eaares(isa) = avg_sim_wca_stg(isa) .LE.
     $            stage_import_wca(isa) + offset(isa,2)
               ENDIF
            ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  operations in wca(s) not rain driven
c
               IF (ngrid_cells_daily(isa) .GT. 0) THEN
                  avg_sim_wca_stg(isa) = 0.0
                  DO i = 1,ngrid_cells_daily(isa)
c
c  compute simulated stages
c
                     depthgw = (ells(node_import_daily(isa,i)) - h
     $               (node_import_daily(isa,i))) * s(node_import_daily
     $               (isa,i))
                     IF (pond(node_import_daily(isa,i)) .GT. depthgw)
     $               THEN
```

```
                        wca_nodal_stage = pond(node_import_daily(isa,i))
     $                  - depthgw + ells(node_import_daily(isa,i))
                     ELSE
                        wca_nodal_stage = (pond(node_import_daily(isa,i)
     $                  ) - depthgw)/s(node_import_daily(isa,i)) + ells
     $                  (node_import_daily(isa,i))
                     ENDIF
                     avg_sim_wca_stg(isa) = avg_sim_wca_stg(isa) +
     $               wca_nodal_stage / ngrid_cells_daily(isa)
                  END DO
c
                  IF (nzone(isa+1) .GT. 0) THEN
                     IF (wca_name(isa) .NE. 'WCA2A') THEN
                        rmax_stg_lok_reg_wca = regstg_wca(isa,
     $                  ibot_zone_indx(isa+1))
                     ELSE
                        IF (avg_sim_wca_stg(iwca3a).LT.regstg_wca
     $                  (iwca3a ,ibot_zone_indx(iwca3a+ 1)) .AND. wcaenv
     $                  (iwca3a) .EQ. 'FALSE') THEN
                           rmax_stg_lok_reg_wca = regstg_wca(isa,
     $                     itop_zone_indx(isa + 1))
                        ELSE
                           rmax_stg_lok_reg_wca = regstg_wca(isa,
     $                     ibot_zone_indx(isa + 1))
                        ENDIF
                     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                     LRGLO(ISA) = avg_sim_wca_stg(isa) .LT.
     $               rmax_stg_lok_reg_wca
                  ENDIF
               ENDIF
            ENDIF
            IF (ngrid_cells_daily_out(isa) .GT. 0) THEN
               stage_outflw_wca(isa) = 0.0
               avg_sim_wca_stg_out(isa) = 0.0
               DO i = 1,ngrid_cells_daily_out(isa)
                  stage_outflw_wca(isa) = stage_outflw_wca(isa) +
     $            daily_stg_import_wca(ngage_index_env(isa,i,2)) /
     $            ngrid_cells_daily_out(isa)
                  depthgw = (ells(node_outflw_daily(isa,i)) - h
     $            (node_outflw_daily(isa,i))) * s(node_outflw_daily
     $            (isa,i))
                  IF (pond(node_outflw_daily(isa,i)) .GT. depthgw) THEN
                     wca_nodal_stage = pond(node_outflw_daily(isa,i)) -
     $               depthgw + ells(node_outflw_daily(isa,i))
                  ELSE
                     wca_nodal_stage = (pond(node_outflw_daily(isa,i)) -
     $               depthgw)/s(node_outflw_daily(isa,i)) + ells
     $               (node_outflw_daily(isa,i))
                  ENDIF
                  avg_sim_wca_stg_out(isa) = avg_sim_wca_stg_out(isa) +
     $            wca_nodal_stage / ngrid_cells_daily_out(isa)
               END DO
            ENDIF
         END DO
      ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
c
c  target is constant
c
         IF (wca_import_variation .EQ. 'CONST') THEN
            DO isa = 1,ntotal_no_import_areas_const
               IF (nmpts_wca(isa) .GT. 0) THEN
                  stage_import_wca(isa) = avg_import_stage(isa)
                  avg_sim_wca_stg(isa) = 0.0
                  DO i = 1,nmpts_wca(isa)
                     depthgw = (ells(node_import(isa,i)) - h(node_import
     $               (isa,i))) * s(node_import(isa,i))
                     IF (pond(node_import(isa,i)) .GT. depthgw) THEN
                        wca_nodal_stage = pond(node_import(isa,i)) -
     $                  depthgw + ells(node_import(isa,i))
                     ELSE
                        wca_nodal_stage = (pond(node_import(isa,i)) -
     $                  depthgw)/s(node_import(isa,i)) + ells
     $                  (node_import(isa,i))
                     ENDIF
                     avg_sim_wca_stg(isa) = avg_sim_wca_stg(isa) +
     $               wca_nodal_stage / nmpts_wca(isa)
                  END DO
                  LWSLO(ISA) = avg_sim_wca_stg(isa).LE.stage_import_wca
     $            (isa) - offset(isa,1)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  IF (nzone(isa + 1) .GT. 0) THEN
                     IF (wca_name(isa) .NE. 'WCA2A') THEN
                        IF (nzone(isa + 1) .LE. 2) THEN
                           rmax_stg_lok_reg_wca = regstg_wca(isa,1)
                        ELSE
                           rmax_stg_lok_reg_wca = regstg_wca(isa,3)
                        ENDIF
                        LRGLO(ISA) = avg_sim_wca_stg(isa) .LT.
     $                  rmax_stg_lok_reg_wca
                     ELSE
                        IF (avg_sim_wca_stg(iwca3a).LT.regstg_wca
     $                  (iwca3a,3))THEN
                           rmax_stg_lok_reg_wca = regstg_wca(isa,3)
                        ELSE
                           rmax_stg_lok_reg_wca = regstg_wca(isa,1)
                        ENDIF
                     ENDIF
                     LRGLO(ISA) = avg_sim_wca_stg(isa) .LT.
     $               rmax_stg_lok_reg_wca
                  ENDIF
               ENDIF
            END DO
         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      RETURN
      END
```

rain_pet_setup.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: rain_pet_setup.F,v 1.9 2005/06/10 13:29:17 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/rain_pet_setup.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE SETS UP RAINFALL AND POTENTIAL EVAPOTRANSPIRATION
C   FOR MODEL DOMAIN
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE rain_pet_setup (day_rain, maxy_nsm, minx_nsm, maxx_nsm,
     & isum, dx, dy, itrib_hydro_cond_opt, numd, mon_pet0, day_pet0,
     & potet, c43pet, c44pet, norstorpet, day_pet0_lok, pet_weight,
     & month, maxy, netzon, nday_yr, rain_def, title_rain, isum_nsm,
     & day_rain_nsm)

      INCLUDE 'wmm_par.inc'

      DIMENSION c43pet(366), c44pet(366), day_pet0(max_ncells,31),
     & day_pet0_lok(31), day_rain(max_ncells,31),
     & day_rain_nsm(max_n_cells_nsm,31), isum(max_n_rows),
     & isum_nsm(max_n_rows_incl_lok), maxx_nsm(max_n_rows_incl_lok),
     & minx_nsm(max_n_rows_incl_lok), mon_pet0(max_ncells),
     & norstorpet(366), numd(12), pet_weight(max_ncells,20),
     & petstn(20), potet(max_ncells), rain_nsm(max_n_cells_nsm)

      CHARACTER*70 title_rain
      CHARACTER*80 snap_tag_rain

      INTEGER rain_def

      REAL mon_pet0, norstorpet
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  READ one month of daily rainfall data to calculate
C  monthly sum (to be used later for ET reductions).

      CALL setgrid(title_rain,maxy_nsm,isum_nsm(maxy_nsm + 1),dx,dy,
     $minx_nsm,maxx_nsm,isum_nsm)
      DO iday = 1,numd(month)
         CALL gread(rain_def, snap_tag_rain, rain_nsm, ierr)
         DO i = maxy_nsm,1,-1
            DO L = isum_nsm(i) + 1,isum_nsm(i + 1)
               day_rain_nsm(L,iday) = rain_nsm(L)
            END DO
         END DO
      END DO

C  map NSM rain to SFWMM rain

      DO iday = 1,numd(month)
         DO i = 1,1168
            day_rain(i,iday) = day_rain_nsm(i,iday)
         END DO
         DO i = 1169, 1195
            day_rain(i,iday) = day_rain_nsm(i + 13,iday)
         END DO
         DO i = 1196, 1222
            day_rain(i,iday) = day_rain_nsm(i + 26,iday)
         END DO
         DO i = 1223, 1250
            day_rain(i,iday) = day_rain_nsm(i + 38,iday)
         END DO
         DO i = 1251, 1278
            day_rain(i,iday) = day_rain_nsm(i + 50,iday)
         END DO
         DO i = 1279, 1306
            day_rain(i,iday) = day_rain_nsm(i + 62,iday)
         END DO
         DO i = 1307, 1334
            day_rain(i,iday) = day_rain_nsm(i + 74,iday)
         END DO
         DO i = 1335, 1362
            day_rain(i,iday) = day_rain_nsm(i + 86,iday)
         END DO
         DO i = 1363, 1391
            day_rain(i,iday) = day_rain_nsm(i + 98,iday)
         END DO
         DO i = 1392, 1420
            day_rain(i,iday) = day_rain_nsm(i + 110,iday)
         END DO
         DO i = 1421, 1449
            day_rain(i,iday) = day_rain_nsm(i + 122,iday)
         END DO
         DO i = 1450, 1478
            day_rain(i,iday) = day_rain_nsm(i + 134,iday)
         END DO
         DO i = 1479, 1507
```

```
            day_rain(i,iday) = day_rain_nsm(i + 146,iday)
         END DO
         DO i = 1508, 1535
            day_rain(i,iday) = day_rain_nsm(i + 159,iday)
         END DO
         DO i = 1536, 1557
            day_rain(i,iday) = day_rain_nsm(i + 178,iday)
         END DO
         DO i = 1558, 1578
            day_rain(i,iday) = day_rain_nsm(i + 198,iday)
         END DO
         DO i = 1579, 1599
            day_rain(i,iday) = day_rain_nsm(i + 218,iday)
         END DO
         DO i = 1600, 1620
            day_rain(i,iday) = day_rain_nsm(i + 238,iday)
         END DO
         DO i = 1621, 1640
            day_rain(i,iday) = day_rain_nsm(i + 258,iday)
         END DO
         DO i = 1641, 1659
            day_rain(i,iday) = day_rain_nsm(i + 279,iday)
         END DO
         DO i = 1660, 1677
            day_rain(i,iday) = day_rain_nsm(i + 301,iday)
         END DO
         DO i = 1678, 1695
            day_rain(i,iday) = day_rain_nsm(i + 323,iday)
         END DO
         DO i = 1696, 1712
            day_rain(i,iday) = day_rain_nsm(i + 345,iday)
         END DO
         DO i = 1713, 1729
            day_rain(i,iday) = day_rain_nsm(i + 367,iday)
         END DO
         DO i = 1730, 1746
            day_rain(i,iday) = day_rain_nsm(i + 389,iday)
         END DO
      END DO
c
c-------------------------------------------------------
c     READ one month of daily Penman-Monteith PET (referenced
c     to grass) for each of the 12 new et basins into petstn array.
c     Then calculate reference et for each NODE (potet array)
c     based on weights. Potet data summed monthly to be used later
c     for ET reductions.
c-------------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      DO I = 1,MAXY
         DO NODE = ISUM(I) + 1, ISUM(I + 1)
            mon_pet0(node) = 0.0
         END DO
      END DO
      DO iday = 1,numd(month)
         nday_yr = nday_yr + 1
         READ (28,'(16X,20F6.3)') (petstn(iz),iz=1,netzon)
         DO I = 1,MAXY
```

```
            DO NODE = ISUM(I) + 1, ISUM(I+1)
               potet(node) = 0.0
               DO j = 2, netzon
                  potet(node) = petstn(j)*pet_weight(node,j-1)
     $            +potet(node)
                  day_pet0(node,iday) = potet(node)
                  mon_pet0(node) = mon_pet0(node) + day_pet0(node,iday)
               END DO
            END DO
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         day_pet0_lok(iday) =  petstn(1)
         c43pet (nday_yr) = petstn(1) / 12.
         c44pet (nday_yr) = petstn(1) / 12.
         norstorpet(nday_yr) = petstn(1)/12.0
         norstorpet(nday_yr) = norstorpet(nday_yr) * 1.0
      END DO
      RETURN
      END
```

readtk.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: readtk.F,v 1.9 2005/06/10 13:29:17 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/readtk.F,v $
C   $Name: rel-5-5 $
C
C    THIS SUBROUTINE READS MID-MONTH CROP COEFFICIENTS FOR EACH
C    LAND USE TYPE AND CONVERTS COEFFICIENTS TO ACTUAL DAY
C    VALUE DEPENDING ON THE RELATIVE POSITION OF THE CURRENT
C    DAY TO TWO MID-MONTH VALUES
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE readtk (ifile, nlu)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'et.inc'

      REAL k(max_n_lu_type,12), midmon(12), tmpk(12)

      DATA midmon /15.0, 46.0, 74.0, 105.0, 135.0, 166.0, 196.0, 227.0,
     & 258.0, 288.0, 319.0, 349.0/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  READ IN THE 'MID-MONTH' (MEAN) CROP COEFFICIENTS

      DO 10 KMONTH = 1,12
         READ(IFILE,50) (K(I,KMONTH), I = 1,NLU)
   10 CONTINUE

C  COMPUTE CROP COEFFICIENTS FOR EACH DAY FOR EACH LAND USE

      DO 40 I = 1,NLU
         DO 20 KMONTH = 1,12
```

```
         TMPK(KMONTH) = K(I,KMONTH)
20    CONTINUE
      DO 30 JDAY = 1,365
         ETK(I,JDAY) = AINTERP(REAL(JDAY),MIDMON,TMPK,12,14.0,16.0,1)
30    CONTINUE
      ETK(I,366) = (ETK(I,365) + ETK(I,1))/ 2.0
40 CONTINUE
   RETURN
50 FORMAT(30F7.2)
   END
```

remov_comm_inp_files.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: remov_comm_inp_files.F,v 1.2 2003/07/03 20:07:42 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/remov_comm_inp_files.F,v $
C   $Name: rel-5-5 $
C
C   Purpose:
C   The main purpose of this subroutine is to remove comment lines from
C   SFWMM input files.  This subroutine determines if an input file is non-
local.
C   If this is the case, then it opens the input file and returns control to the
C   calling program. Otherwise, it removes comment lines from the original input
C   file and writes a cleaned version file in the unit number specified in the
C   ALTWMM file, so that the file can be read again in the calling subroutine of
C   the SFWMM code.
C
C   Comments can be inserted into an ASCII input data file using the following
C   rules:
C   1) A line must have a '!' or '*' or '#' in the first column.
C   '!' ==> denotes a comment line (=ignore)
C   '*' ==> denotes a comment line (=ignore)
C   '#' ==> denotes a comment line (=ignore)
C   2) A group or block of comments can be created by starting the line, in
C   column 1, with the keyword BEGCOMM (upper case, lower case or any
C   combination).
C   The same group of comments is ended by starting the line, in column 1, with
C   the keyword ENDCOMM (upper case, lower case or any combination).
C
C   Warning:  These rules need to be followed exactly.  Otherwise the SFWMM
model
C   will crash or produce wrong results due to misalignment in the input files.
C
C   Notes:
C   - This subroutine is transparent to the user
C   - The documented file is preserved in the simulation input directory
C   - A copy of the cleaned version of the file is created in the simulation
```

```
C     input directory using the same name followed by the ".cln" extension
C   - The cleaned version of the file is used in the calling subroutine
C     of the code
C   - A copy of the documented file is created in the back-up under the
C     simulation output directory once the model execution completes normally or
C     the user terminated the execution (?????)
C
C                INPUT VARIABLES; MAIN PROGRAM
C
C       infile      Character*200, path of the input file name
C       iunit2      Unit number of a file to be opened
C
C-----------------------------------------------------------------------------

      subroutine remov_comm_inp_files(infile,iunit2)

      parameter(iunitdum=333)
      character*200 infile
      integer nlines
      integer iunit2,indx
c
c     Returns the starting position of the substring '/' within a string to
detect
c     if file is non-local.
c
      indx=0
      indx=index(infile,'/')
c
c     If the file is not in the local directory, then open file and return.
c     File is connected to the unit number specified in the ALTWMM input file
c
      if(indx.gt.0)then
        open(iunit2,file=infile,status='old')
        return
      endif
c
c     A temporal unit number, iunitdum, for reading from the documented input
c     file is opened
c
      open(iunitdum,file=infile,status='old')
c
c     Find number of lines in the file connected to iunitdum unit
c
      call find_num_lines(iunitdum,nlines)
c
c     Remove comment lines and write cleaned file in the unit number specified
c     in the ALTWMM file
c
      call remove_comment(infile,iunit2,iunitdum,nlines)

      END

C-----------------------------------------------------------------------------
C     Purpose:
C     This subroutine finds the number of lines in a file connected to
C     iunitdum unit
C
```

```
C       Input variable:
C       iunitdum    Temporal unit number for reading from the documented input
C                   file
C
C       Output variable:
C       nlines      Number of lines in the file connected to iunitdum unit
C-----------------------------------------------------------------------------
      subroutine find_num_lines(iunitdum,nlines)

      character*1 char_one
      integer nlines

      nlines=0
      icond=1
      do while (icond.gt.0)
         read(iunitdum,'(a)',end=150)char_one
         nlines=nlines+1
      end do
 150  icond=0

      END

C-----------------------------------------------------------------------------
C     Purpose:
c     This subroutine removes comment lines and writes cleaned file in the unit
c     number specified in the ALTWMM file
C
C       Input variables:
C       infile      Character*200, path of the documented input file
C       iunit2      Unit number for writing to the cleaned version input file
C       iunitdum    Temporal unit number connected to the file being opened
C       nlines      Number of lines in the file connected to iunitdum unit
C       Output variable:
C       cleanname   Character*200, path of the the cleaned version input file
C
C-----------------------------------------------------------------------------
      subroutine remove_comment(infile,iunit2,iunitdum,nlines)

      character*200 infile,cleanname
      integer iunit2
      character*1 char_one, bang, star, pound
      character*1 char_BEGCOMM(7),char_ENDCOMM(7)
      character*3000 line
      dimension line_size(nlines)

      data bang, star, pound /'!','*','#'/
      data char_BEGCOMM/'B','E','G','C','O','M','M'/
      data char_ENDCOMM/'E','N','D','C','O','M','M'/
c
c    Open new file connected to the unit number specified in the ALTWMM file
c    Unix function, LNBLNK, returns the index of the last non-blank character
c    in a String.
c
      cleanname=infile(1:LNBLNK(infile))//'.cln'
      open(iunit2,form='formatted',file=cleanname,status='unknown')

      rewind(iunitdum)
```

```
c
c     Find size of each line in file
c
      do i=1,nlines
         char_one=' '
         j=0
         do while (char_one.ne.'\n')
c
c       Unix function that reads a single character into "char_one" in stream
c         mode from iunitdum (by-passing normal formatted output) using getc(3);
c         Detectes the end of line character;
c         Returns in "iostatus" 0 on success, -1 on end-of-file, and
c         the error code from ferror(3) otherwise.
c
            call FGetC(iunitdum,char_one,iostatus)
            j=j+1
         end do
         line_size(i)=j-1
      end do

      rewind(iunitdum)
c
c     Assigns flag, in_com, equal to zero.  This flag indicates the start or end
c     of a block of comments
c
      in_com=0
c
c     Check for comments in a line
c
      do i=1,nlines
        read(iunitdum,'(a)',end=300)line
c
c       Check if a file line starts with the keyword BEGCOMM (upper case, lower
c       case or any combination).  If this is the case, it is the start of a
c       a block of comments. Then, it makes in_com = 1
c
        call find_comment(char_BEGCOMM,line,in_com,1)
c
c       If this is not a comment line we have found real data.  In that case,
c       write line in the file so that the line will be read again in the code.
c       The conditions for a comment line are as follows:
c       in_com = 1, or line have a '!' or '*' or '#' in the first column
c
        if(in_com.ne.1.and.line(1:1).ne.bang.and.line(1:1).ne.star.and.
     1                line(1:1).ne.pound)then
            write(iunit2,'(a)')line(1:line_size(i))
        endif
c
c       Check if a file line starts with the keyword ENDCOMM (upper case, lower
c       case or any combination).  If this is the case, it is the end of a
c       a block of comments. Then, it makes in_com = 0
c
        call find_comment(char_ENDCOMM,line,in_com,0)
      end do
c
c     Close temporal unit number and rewind iunit2 unit number so that the file
c     can be read again in the code
```

```
c
300   close(iunitdum)
      rewind(iunit2)

      END

C-----------------------------------------------------------------------------
C   Purpose:
C   This subroutine checks the first seven ASCII elements of a file line
C
C       Input variables:
C       char_com  Character*1(7), data list of characters that are used to
C                 compare the first seven characters of line
C       line      Character*3000,contents of a record that has been read from
the
C                 input file
C       iflag     Value given to in_com flag
C
C       Output variable:
C       in_com    Flag indicating the start or end of a block of comments
C                 =1,indicates the start of a block of comments
C                 =0,indicates the end of a block of comments
C
C-----------------------------------------------------------------------------
      subroutine find_comment(char_com,line,in_com,iflag)
      character*3000 line
      character*1 char_com(7)
      integer code_low,code_up,in_com,iflag

      do j=1,7
c
c      ASCII element 65 marks the start of upper case characters and 97 marks
c      the start of lower case characters. Then, the ASCII element for a lower
c      case character is the ASCII element for an upper case character plus 32.
c
       code_up=ichar(char_com(j))
       code_low=ichar(char_com(j))+32
c
c      Check if the ASCII element of a single character of a line, in a specific
c      position, is equal to the ASCII element of a single character specified
c      in a data statement, respectively.  If each of the first seven ASCII
c      elements meets the condition, assigns in_com equal to iflag; otherwise,
c      returns.
c
       if(ichar(line(j:j)).eq.code_low.or.
     1  ichar(line(j:j)).eq.code_up)then
        if(j.eq.7)in_com=iflag
       else
        return
       endif
      enddo
      end
```

resadj.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C                                                                      -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: resadj.inc,v 1.6 2004/01/23 14:12:32 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/resadj.inc,v $
C       $Name: rel-5-5 $
-
C                                                                      -
C                                                                      -
C-----------------------------------------------------------------------
      common /resadj/ sum_chg_dpth_ovl_flw(max_ncells),
     &daily_ET(max_ncells),daily_outf_fc_frm_res_to_cell(max_ncells),
     &daily_outf_seep_loc_to_res_cell(max_ncells),
     &daily_chg_dpth_ovl_vol(max_ncells),
     &pond_adj_due_to_res(max_ncells),daily_excess_adjust(max_ncells),
     &pond_wr(max_ncells),adjust_dur_excess(max_ncells),
     &adjust_dur_deficit(max_ncells),
     &daily_outf_from_res_to_cell_max(max_ncells),
     &rinit_res_stg(max_n_res),prev_adjust_for_area(max_ncells),
     &head_cell_beg(max_ncells),daily_chg_dpth_ovl_vol1(max_ncells),
     &daily_ET_outside_res(max_ncells),
     &daily_adjust_for_max_vol(max_ncells),
     &daily_gw_loss_cell(max_ncells),seep_loss_res_cell(max_ncells),
     &sum_chg_depth_levee_spg_unadj(max_ncells),
     &SUMET_outside_res(max_ncells),
     &daily_inflow_to_res_frm_cnl(max_n_res),
     &est_cell_stg_o_res(max_ncells),
     &width_of_res(max_n_res),frac_seep_max_lim(max_n_res),
     &noresincell(max_ncells),
     &icell_trg_loc(max_n_res,max_nostr_res,max_n_out_loc_str),
     &ITRGCNL(max_n_res,max_nostr_res,max_n_out_loc_str),
     &NODEPOS_RCNL_TRG(max_n_res,max_nostr_res,max_n_out_loc_str),
     &ires_index_in_cell(max_ncells,max_n_res_in_cell),
     &frac_seep_dir_cnl(max_n_res),adjust_dur_deficit0(max_ncells),
     &ires_lev_seep_dir_cnl_opt(max_n_res)
      character ires_lev_seep_dir_cnl_opt*3
C-----------------------------------------------------------------------
```

resasr_sim.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: resasr_sim.F,v 1.8 2005/06/10 13:29:17 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/resasr_sim.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine simulates the proposed reservoir/ASR systems
C   in the Caloosahatchee and St. Lucie basins.
C
C   hres0 <--- beginning of day reservoir water level
C   asr_vol0 <--- beginnning of day asr bubble volume
C   hres0 <--- 0:ASR not yet used, 1:ASR injection, 2:ASR recovery
C        = 0
C   qin_max = qout_max <--- beginning of day reservoir pump caps
C                     = 99999. cfs
C   injection_max0 = withdrawal_max0 <--- beginning of day asr pump caps
C           = 500 MGD = 773.61 cfs
C
C  ***      ALL CALCULATIONS DONE IN UNITS OF CFS-DAY, UON !!!
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE resasr_sim (ibasin, iyr, month, iday, ndays, iopt_asr,
     & iasrenv, bsn2est_af, dmnd0_day, roff0_day, rem_supp0_day,
     & bflo_cap0_mo, basin_rf_day, basin_et_day, ares, hres_min,
     & hres_max, hres0, asr_vol0, in_out_asr, qin_max0, qout_max0,
     & injection_max0, withdrawal_max0, res_asr_outflow, stglok, util,
     & rmin_lok_stg_del_est, opt_bsn_prio_uncond, lok_split_option,
     & stagelo_res, n_fc_zones)

      CHARACTER opt_bsn_prio_uncond*3

      LOGICAL iasrenv, iopt_asr
```

```
      REAL injection_max, injection_max0, qin_max, qin_max0, qout_max,
     & qout_max0, withdrawal_max, withdrawal_max0

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  input variables:
C       opt_bsn_prio_uncond = YES   ! meet basin demand first
C       iopt_asr = 1   ! use ASR option
C       iasrenv = 1    ! use ASR to meet estuarine requirement
C
C  Caloosahatchee estuary demands were obtained from R. Chamberlain
C  (10/22/93 memo)
C      April, ..., October:  750 cfs  (1487.625 ac-ft/day)
C      November, ..., March: 500 cfs  (991.750 ac-ft/day)
C
C  St. Lucie estuary demands were obtained from D. Haunert (3/15/93 memo)
C  old stuff
C      April: 500 cfs  (991.71 ac-ft/day)
C      May:   600 cfs  (1190.06 ac-ft/day)
C      June:  700 cfs  (1388.40 ac-ft/day)
C      July:  700 cfs  (1388.40 ac-ft/day)
C  new stuff
C      January to December: 350 cfs  (694.225 ac-ft/day)
C
C  assume reservoir properties
C    let: constant reservoir seepage loss rate = .0114 ft per day

      res_seepage = .0114
      qin_max = qin_max0
      qout_max = qout_max0
c assume asr properties
      withdrawal_max = withdrawal_max0
      injection_max = injection_max0
c
c      WRITE(87,20)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c initialize some variables; be sure to convert numbers to cfs-day
      rem_ro = 0.0
      rem_dmnd = 0.0
      ro_to_lok_or_est = 0.0
      bsn2est = bsn2est_af / 1.9835
      rem_supp0 = rem_supp0_day * (ndays - iday + 1)
      rem_supp = rem_supp0 / 1.9835
      bflo_cap_mo = bflo_cap0_mo / 1.9835
      dmnd_day = dmnd0_day / 1.9835
      roff_day = roff0_day / 1.9835
c initialize reservoir variables
      bsn2res = 0.0
      res2est = 0.0
      res2bsn = 0.0
      actual_et = 0.0
      actual_seepage = 0.0
      spill_over = 0.0
      spil2supp = 0.0
c initialize asr variables
      asr2bsn = 0.0
```

```
      asr2est = 0.0
      res2asr = 0.0
      asr_injection_loss = 0.0
      asr_storage_loss = 0.0
      decay_rate = 0.0
c      util = 0.75
c note: recovery rate of 60 %/yr == decay rate of 40%/yr or 0.39854%/day
c       assume a utilization of 0.75
c       decay factor = loss term during storage
c       recovery efficiency  = could be utilization ???
c update bubble size based on decay rate concept
      asr_vol = asr_vol0 / 1.9835 * (1. - decay_rate / 100.)
      asr_storage_loss = asr_vol0 / 1.9835 * (decay_rate / 100.)
c
c update water level in reservoir based on rf-et-seepage
      hres = hres0 + basin_rf_day - basin_et_day - res_seepage
      actual_et = basin_et_day
      actual_seepage = res_seepage
      IF (hres .LT. 0.) THEN
c calculate et and seepage to bring hres to zero
         h0_plus_rf = hres0 + basin_rf_day
         IF (h0_plus_rf .GT. basin_et_day) THEN
            actual_et = basin_et_day
            actual_seepage = h0_plus_rf - actual_et
         ELSE
            actual_et = h0_plus_rf
            actual_seepage = 0.0
         ENDIF
         hres = 0.0
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      hres1 = hres
      IF (hres .GT. hres_max) THEN
         spill_over = (hres - hres_max)* ares / 1.9835
         hres = hres_max
c note: spill_over are assumed to be not subject to structure
c       capacity constraints, all of which are assumed to GO TO estuary
c       and not to the basin
         IF (rem_supp .LE. spill_over) THEN
c all of remaining supplemental estuarine requirement will be met by
c all or part of reservoir spill_over
            res2est = rem_supp
            spill_over = spill_over - rem_supp
            rem_supp = 0.0
         ELSE
c all reservoir spill_over will be used to meet part of remaining
c supplemental estuarine requirement
            res2est = spill_over
            rem_supp = rem_supp - spill_over
            spill_over = 0.0
         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      spil2supp = res2est
      hres_strt = hres
c res2est = total (spill over + regular) reservoir release used
c     exclusively to meet estuarine demands
```

```
c spill_over = spill over that is not used to meet estuarine demands
c spil2supp = spill over that is used to meet estuarine dmds
c           = subset of res2est
      supp_day = rem_supp / (ndays - iday + 1)
      supp_to_est_day = supp_day
      unmet_supp = supp_to_est_day
c
      IF (roff_day .GT. 0.) THEN
c first, use reservoir to meet estuarine supplemental requirement and
c THEN, fill reservoir with remaining runoff
         IF (supp_to_est_day .GT. 0.) THEN
c all of today's runoff goes to the estuary,i.e. bsn2est=roff_day
c release water from detention basin/reservoir to meet estuarine demand
c note: bflo_cap_mo is calculated after basin runoff is released
c       to estuary
            avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
            est_release = amin1(qout_max,supp_to_est_day ,avail_res_rel)

            hres = hres - est_release * 1.9835 / ares
            IF (est_release .EQ. supp_to_est_day) THEN
c all supplemental requirement for today will be met by reservoir
               supp_to_est_day = 0.0
               unmet_supp = 0.0
            ELSE
c not all supplemental requirement for today will be met by reservoir
               supp_to_est_day = supp_to_est_day - est_release
               unmet_supp = supp_to_est_day
            ENDIF
c other sources of supplement is necessary IF unmet_supp > 0.0
            rem_supp = rem_supp - supp_day + unmet_supp
c or rem_supp = rem_supp - est_release
c        where: est_release = supp_day - unmet_supp
c.....RES2EST:
            res2est = res2est + est_release
c       meet remaining estuarine requirement IF asr option is on and
c       is allowed to meet environmental needs
            IF (iopt_asr .AND. iasrenv .AND. unmet_supp .GT. 0.) THEN
c.....ASR2EST:
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               asr2est = amin1(withdrawal_max,supp_to_est_day,asr_vol)
clmb       IF (asr2est.EQ.withdrawal_max) THEN
clmb         PRINT *,1
clmb       ELSE IF(asr2est.EQ.supp_to_est_day) THEN
clmb         PRINT *,2
clmb       ELSE
clmb         PRINT *,3
clmb       ENDIF
               asr_vol = asr_vol - asr2est
               IF (asr2est .EQ. supp_to_est_day) THEN
c all remaining supplemental requirement for today can be met by the asr
                  supp_to_est_day = 0.0
                  unmet_supp = 0.0
               ELSE
                  supp_to_est_day = supp_to_est_day - asr2est
                  unmet_supp = supp_to_est_day
               ENDIF
               IF (asr2est .GT. 0.) in_out_asr = 2
```

```
c lake supplement is necessary IF unmet_supp > 0
               rem_supp = rem_supp - asr2est
            ENDIF
         ENDIF
         rem_ro = roff_day - bsn2est
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c FILL UP ASR WELL FIRST
c DO asr injection IF water is available from the reservoir
c       and asr well not yet used to supplement reservoir in
c       meeting estuarine demand; Assumption: Reservoir release
c       does not exceed remaining runoff for the day !!
         IF (iopt_asr .AND. in_out_asr .EQ. 0) THEN
            avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
c.....RES2ASR:
            res2asr = amin1(injection_max,avail_res_rel,rem_ro)
            hres = hres - res2asr * 1.9835 / ares
            injection_max = injection_max - res2asr
            asr_inj_loss = res2asr * (1. - util)
            asr_vol = asr_vol + (res2asr - asr_inj_loss)
            asr_injection_loss = asr_inj_loss
            IF (res2asr .GT. 0.) in_out_asr = 1
         ENDIF
c remaining runoff is portion of original runoff not used to
c meet estuarine requirement and can be used to fill detention basin
c FILL UP RESERVOIR NEXT
         avail_res_sto = amax1(0.,hres_max-hres)* ares / 1.9835
c.....BSN2RES:
         bsn2res = amin1(qin_max,bflo_cap_mo,avail_res_sto,rem_ro)
         IF (ibasin .EQ. 1) THEN
            IF (stglok .LT. 11.1) bsn2res = 0.0
         ELSE
            IF (lok_split_option .EQ. 1) THEN
               stglokup = stagelo_res
               stg_ref = 14.5
            ELSE
               stglokup = stglok
               CALL bpts(iday,month,10.0,1,n_fc_zones,1,stg_ref)
            ENDIF
            stg_backflow = amin1(14.5,stg_ref)
            IF (stglokup.LT.stg_backflow) bsn2res = 0.0
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         hres = hres + bsn2res * 1.9835 / ares
         rem_ro = rem_ro - bsn2res
         ro_to_lok_or_est = rem_ro
         bflo_cap_mo = bflo_cap_mo - bsn2res
c note: ro_rev = ro_old - bsn2est - bsn2res
c       ro_rev >= ro_to_lok_or_est
      ELSE IF(supp_to_est_day .GT. 0.0 .OR. dmnd_day .GT. 0.0) THEN
         IF (stglok .GE. rmin_lok_stg_del_est .OR. opt_bsn_prio_uncond
     $   .EQ. 'YES') THEN
c meet local basin demands before estuarine demands
            IF (dmnd_day .GT. 0.0) THEN
c satisfy basin demand based on available reservoir release
c (i.e. storage above dead storage)
c meet basin demand by making reservoir release
               avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
```

```
c.....RES2BSN:
               res2bsn = amin1(qout_max,dmnd_day,avail_res_rel)
               hres = hres - res2bsn * 1.9835 / ares
               IF (res2bsn .EQ. dmnd_day) THEN
c all basin demand met by reservoir
                  dmnd_day = 0.0
               ELSE
c not all basin demand met by reservoir, other sources still have to
c meet portion of it
c dmnd_day becomes unmet basin demand
                  dmnd_day = dmnd_day - res2bsn
               ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               IF (iopt_asr .AND. dmnd_day .GT. 0.0) THEN
c.....ASR2BSN
                  asr2bsn = amin1(withdrawal_max,dmnd_day ,asr_vol)

                  asr_vol = asr_vol - asr2bsn
c update max withdrawal rate for asr pumping to estuary, IF necessary
                  withdrawal_max = withdrawal_max - asr2bsn
                  IF (asr2bsn .EQ. dmnd_day)THEN
c all remaining basin demand met by asr
                     dmnd_day = 0.0
                  ELSE
c not all basin demand met by asr, lake still has to meet portion of it
c dmnd_day becomes unmet basin demand
                     dmnd_day = dmnd_day - asr2bsn
                  ENDIF
                  IF (asr2bsn .GT. 0.) in_out_asr = 2
               ENDIF
            ENDIF
c
            IF (supp_to_est_day .GT. 0.0) THEN
c release water from detention basin/reservoir to meet estuarine demand
               avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
               est_release = amin1(qout_max,supp_to_est_day ,
     $         avail_res_rel)
               hres = hres - est_release * 1.9835 / ares
               IF (est_release.EQ.supp_to_est_day) THEN
c all supplemental requirement for today will be met by reservoir
                  supp_to_est_day = 0.0
                  unmet_supp = 0.0
               ELSE
c not all supplemental requirement for today will be met by reservoir
                  supp_to_est_day = supp_to_est_day - est_release
                  unmet_supp = supp_to_est_day
               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c other sources of supplement is necessary IF unmet_supp > 0.0
               rem_supp = rem_supp - supp_day + unmet_supp
c or rem_supp = rem_supp - est_release
c        where: est_release = supp_day - unmet_supp
c.....RES2EST:
               res2est = res2est + est_release
c       meet remaining estuarine requirement IF asr option is on and
c       is allowed to meet environmental needs
```

```
               IF (iopt_asr .AND. iasrenv .AND. unmet_supp .GT. 0.)THEN
c.....ASR2EST:
                  asr2est = amin1(withdrawal_max,supp_to_est_day ,
     $            asr_vol)
clmb       IF (asr2est.EQ.withdrawal_max) THEN
clmb         PRINT *,1
clmb       ELSE IF(asr2est.EQ.supp_to_est_day) THEN
clmb         PRINT *,2
clmb       ELSE
clmb         PRINT *,3
clmb       ENDIF
                  asr_vol = asr_vol - asr2est
                  IF (asr2est .EQ. supp_to_est_day) THEN
c all remaining supplemental requirement for today can be met by the asr
                     supp_to_est_day = 0.0
                     unmet_supp = 0.0
                  ELSE
                     supp_to_est_day = supp_to_est_day - asr2est
                     unmet_supp = supp_to_est_day
                  ENDIF
                  IF (asr2est .GT. 0.) in_out_asr = 2
c lake supplement is necessary IF unmet_supp > 0
                  rem_supp = rem_supp - asr2est
               ENDIF
c
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         ELSE
c meet estuarine demands before local basin demands
            IF (supp_to_est_day .GT. 0.0) THEN
c release water from detention basin/reservoir to meet estuarine demand
               avail_res_rel = amax1(0.,hres-hres_min)* ares / 1.9835
               est_release = amin1(qout_max,supp_to_est_day ,
     $         avail_res_rel)
               hres = hres - est_release * 1.9835 / ares
               IF (est_release .EQ. supp_to_est_day) THEN
c all supplemental requirement for today will be met by reservoir
                  supp_to_est_day = 0.0
                  unmet_supp = 0.0
               ELSE
c not all supplemental requirement for today will be met by reservoir
                  supp_to_est_day = supp_to_est_day - est_release
                  unmet_supp = supp_to_est_day
               ENDIF
c other sources of supplement is necessary IF unmet_supp > 0.0
               rem_supp = rem_supp - supp_day + unmet_supp
c or rem_supp = rem_supp - est_release
c        where: est_release = supp_day - unmet_supp
c.....RES2EST:
               res2est = res2est + est_release
c       meet remaining estuarine requirement IF asr option is on and
c       is allowed to meet environmental needs
               IF (iopt_asr .AND. iasrenv .AND. unmet_supp .GT. 0.) THEN
c.....ASR2EST:
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  asr2est = amin1(withdrawal_max,supp_to_est_day ,
     $            asr_vol)
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
clmb       IF (asr2est.EQ.withdrawal_max) THEN
clmb         PRINT *,1
clmb       ELSE IF(asr2est.EQ.supp_to_est_day) THEN
clmb         PRINT *,2
clmb       ELSE
clmb         PRINT *,3
clmb       ENDIF
                  asr_vol = asr_vol - asr2est
c update max withdrawal rate for asr pumping to basin, IF necessary
                  withdrawal_max = withdrawal_max - asr2est
                  IF (asr2est .EQ. supp_to_est_day) THEN
c all remaining supplemental requirement for today can be met by the asr
                     supp_to_est_day = 0.0
                     unmet_supp = 0.0
                  ELSE
                     supp_to_est_day = supp_to_est_day - asr2est
                     unmet_supp = supp_to_est_day
                  ENDIF
                  IF (asr2est .GT. 0.) in_out_asr = 2
c lake supplement is necessary IF unmet_supp > 0
                  rem_supp = rem_supp - asr2est
               ENDIF
c
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
            IF (dmnd_day .GT. 0.0) THEN
c satisfy basin demand based on available reservoir release
c (i.e. storage above dead storage)
c meet basin demand by making reservoir release
               avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
c.....RES2BSN:
               res2bsn = amin1(qout_max,dmnd_day,avail_res_rel)
               hres = hres - res2bsn * 1.9835 / ares
               IF (res2bsn.EQ.dmnd_day) THEN
c all basin demand met by reservoir
                  dmnd_day = 0.0
               ELSE
c not all basin demand met by reservoir, other sources still have to
c meet portion of it
c dmnd_day becomes unmet basin demand
                  dmnd_day = dmnd_day - res2bsn
               ENDIF
c
               IF (iopt_asr .AND. dmnd_day .GT. 0.0) THEN
c.....ASR2BSN:
                  asr2bsn = amin1(withdrawal_max,dmnd_day,asr_vol)
                  asr_vol = asr_vol - asr2bsn
                  IF (asr2bsn .EQ. dmnd_day) THEN
c all remaining basin demand met by asr
                     dmnd_day = 0.0
                  ELSE
c not all basin demand met by asr, lake still has to meet portion of it
c dmnd_day becomes unmet basin demand
                     dmnd_day = dmnd_day - asr2bsn
                  ENDIF
```

```
                  IF (asr2bsn .GT. 0.) in_out_asr = 2
               ENDIF
c
            ENDIF
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      ELSE
c there's no basin runoff, no supplemental req't to estuary and
c no demand for this day
c       No asr withdrawal is necessary and no asr injection from reservoir
c       is done since no net runoff exists for this case.
         CONTINUE
      ENDIF
      dstostrt = (hres-hres_strt) * ares / 1.9835
clb      WRITE(87,30) spill_over,roff0_day/1.9835,supp_day
clb     &           ,bsn2est,bsn2res,ro_to_lok_or_est
clb     &           ,bflo_cap0_mo/1.9835,bflo_cap_mo,res2bsn
c output for unit 87 mostly in cfs
      IF (ibasin .EQ. 1) THEN
         WRITE(87,20) iyr,month,iday,spill_over,roff0_day  /1.9835,
     $   supp_day ,bsn2est,bsn2res,ro_to_lok_or_est ,dmnd0_day / 1.9835,
     $   dmnd_day,res2bsn,res2est,unmet_supp,spil2supp ,hres_strt,hres,
     $   dstostrt,res2asr
      ELSE
         WRITE(88,20) iyr,month,iday,spill_over,roff0_day / 1.9835,
     $   supp_day ,bsn2est,bsn2res,ro_to_lok_or_est ,dmnd0_day / 1.9835,
     $   dmnd_day,res2bsn,res2est,unmet_supp,spil2supp ,hres_strt,hres,
     $   dstostrt,res2asr
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c output for unit 88 in cfs
c         WRITE(88,'(3i5,f10.3)') iyr,month,iday,rem_supp
casr         IF (iday.EQ.ndays) WRITE(88,'(3i5,f10.3)')
casr     &                           iyr,month,iday,rem_supp
c         IF (iday.EQ.ndays) WRITE(87,20)
c
      tot_res_loss = (actual_et+actual_seepage)*ares
c output for unit 89 mostly in acre-ft
      IF (ibasin .EQ. 1) THEN
         WRITE(89,'(i5,2i3,2x,10f10.3)')iyr,month,iday,basin_rf_day*ares
     $   ,basin_et_day*ares,actual_et*ares ,res_seepage*ares
     $   ,actual_seepage*ares ,tot_res_loss,hres0,hres1

      ELSE
         WRITE(91,'(i5,2i3,2x,10f10.3)')iyr,month,iday,basin_rf_day*ares
     $   ,basin_et_day*ares,actual_et*ares ,res_seepage*ares
     $   ,actual_seepage*ares ,tot_res_loss,hres0,hres1

      ENDIF
c output for unit 84 all in cfs (or cfs-day)
      IF (iopt_asr) THEN
         IF (ibasin .EQ. 1) THEN
            WRITE(84,'(i5,2i3,2x,10f15.3)')iyr,month,iday,asr_vol0
     $      / 1.9835,asr_vol,asr_injection_loss,asr_storage_loss,
     $      res2asr,asr2bsn,asr2est

         ELSE
```

```
            WRITE(86,'(i5,2i3,2x,10f15.3)')iyr,month,iday,asr_vol0
     $      / 1.9835,asr_vol,asr_injection_loss,asr_storage_loss,
     $      res2asr,asr2bsn,asr2est

         ENDIF
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c  asr_vol0  = volume of asr bubble at the beginning of the day
c  asr_vol   = volume of asr bubble at the END of the day
c  hres0     = beginning of day stage
c  hres1     = reservoir stage after rf, et & seepage but before spill
c  hres_strt = reservoir stage after spill
c  hres      = END of day reservoir stage
c
c30    FORMAT(12f10.3,2f7.3,2f10.3))
c
c be sure to convert numbers to cfs-day
      res_asr_outflow = (res2est+asr2est+spill_over) * 1.9835
      hres0 = hres
      qin_max0 = qin_max
      qout_max0 = qout_max
      asr_vol0 = asr_vol * 1.9835
      injection_max0 = injection_max
      withdrawal_max0 = withdrawal_max
      rem_supp0_day = rem_supp * 1.9835/(ndays-iday+1)
      bflo_cap0_mo = bflo_cap_mo * 1.9835
      roff0_day = rem_ro * 1.9835
      dmnd0_day = dmnd_day * 1.9835
c
      RETURN
   10 FORMAT('spill_over    bsn_ro  supp_day  bsn2est  bsn2res  exss_ro
     $bsn_dmd1   bsn_dmd   res2bsn   res2est  unmtsupp spil2supp h_strt
     $ h_fin  dstostrt   res2asr')
   20 FORMAT(I5,2I3,2x,12f10.3,2f7.4,2f10.3)
   30 FORMAT(f10.3,'    bsn_ro  supp_day  bsn2est  bsn2res  exss_ro bflo
     $cap0   bflocap   res2bsn   res2est  unmtsupp spil2supp h_strt  h_f
     $in  dstostrt   res2asr')
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

resbudg.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: resbudg.inc,v 1.4 2003/07/03 20:08:03 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/resbudg.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /resbudg/ res_rain(max_n_res),res_et(max_n_res),
     &res_struc_inflw(max_n_res),res_struc_outflw(max_n_res),
     &res_seep_out(max_n_res),res_lvseep_out(max_n_res),
     &prev_stage_res(max_ncells,max_n_res_in_cell),
     &res_to_asr(max_n_res),res_from_asr(max_n_res),
     &res_name_budg(max_n_sm_res)
C-----------------------------
      CHARACTER res_name_budg*6
```

reserv_cell_equil_stg.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: reserv_cell_equil_stg.F,v 1.6 2003/07/03 20:07:43 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/reserv_cell_equil_stg.F,v $
C   $Name: rel-5-5 $
C
C
C   PURPOSE OF SUBROUTINE IS TO COMPUTE THE WEIGHTED MEAN
C   WATER SURFACE ELEVATION FOR A GRID CELL CONTAINING  SMALL
C   RESERVOIR(S) WHICH ARE TREATED INDEPENDENTLY. THIS MEAN ELEV.
C   IS NEEDED FOR CALCULATING MAXIMUM LIMIT TO SEEPAGE LOSS
C   FROM RESERVOIR TO CELL SO THAT NO REVERSAL OF GRADIENT
C   BETWEEN HEAD IN GRID CELL AND HEAD IN RESERVOIR OCCURS.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE reserv_cell_equil_stg (stage_res, ells_in_res, ells,
     & sfactor, total_small_afact, idnstrm, h, pond, s, equil_stg, ia,
     & ir, i, carfact)

      INCLUDE 'wmm_par.inc'

      DIMENSION ells(max_ncells),
     & ells_in_res(max_ncells,max_n_res_in_cell),
     & h(max_ncells),pond(max_ncells), idnstrm(max_n_res,max_nostr_res),
     & s(max_ncells), sfactor(max_n_res),carfact(max_ncells),
     & stage_res(max_ncells,max_n_res_in_cell),
     & total_small_afact(max_ncells)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

       stage_cell = h(idnstrm(ia,i)) + pond(idnstrm(ia,i))
       frac_rem_cell_area = 1.0 - total_small_afact(idnstrm(ia,i))
     &  - carfact(idnstrm(ia,i))
```

```
       IF (ells_in_res(idnstrm(ia,i),ir) .LE.
     & ells(idnstrm(ia,i))) THEN

C land surface elevation in reservoir less than or equal to mean
C surface elevation within grid cell outside seservoir

C equivalent depth of water in reservoir above mean land surface
C elevation in grid cell outside reservoir

          rsdpth1 = amax1(stage_res(idnstrm(ia,i),ir)
     &    - ells(idnstrm(ia,i)),0.0)*sfactor(ia)

C equivalent depth of available storage in grid cell outside reservoir

          dpth_to_wt_cell = (ells(idnstrm(ia,i))
     &    - h(idnstrm(ia,i)))*frac_rem_cell_area*s(idnstrm(ia,i))
     &    - pond(idnstrm(ia,i))*frac_rem_cell_area
c         print *,rsdpth1,dpth_to_wt_cell,total_small_afact
c    &     (idnstrm(ia,i)),stage_res(idnstrm(ia,i),ir),'  CDCDCD  '
c    &     ,stage_cell
          IF (rsdpth1 .GE. dpth_to_wt_cell) THEN
c equivalent depth of water in reservoir above mean land surface of
c grid cell is GREATER than depth of available storage in grid cell
                equil_stg = (stage_res(idnstrm(ia,i),ir)
     &          *sfactor(ia) + amax1(stage_cell,ells
     &          (idnstrm(ia,i)))*frac_rem_cell_area)
     &          /(sfactor(ia) + frac_rem_cell_area)
          ELSE
c equivalent depth of water in reservoir above mean land surface of
c grid cell is LESS than depth of available storage in grid cell.
c MUST recompute equivalent depth of storage in reservoir above
c the reservoir bottom since we know the eqilibrium (mean)
c stage for reservoir-grid cell system is going to be less
c than mean land surface elevation for area outside reservoir
             rsdpth1 = amax1(stage_res(idnstrm(ia,i),ir)
     &       - ells_in_res(idnstrm(ia,i),ir),0.0)*sfactor(ia)
             dpth_to_wt_cell = (ells_in_res(idnstrm(ia,i),ir)
     &       - h(idnstrm(ia,i)))*s(idnstrm(ia,i))
     &       * frac_rem_cell_area
             IF (rsdpth1 .GE. dpth_to_wt_cell) THEN
c equivalent depth of water in reservoir above reservoir bottom is
c GREATER than the eqivalent available storage in area outside reservoir
c below reservoir bottom
                equil_stg = (stage_res(idnstrm(ia,i),ir)
     &          *sfactor(ia) + h(idnstrm(ia,i))*s(idnstrm(ia,i))
     &          *frac_rem_cell_area)/(sfactor(ia)
     &          +s(idnstrm(ia,i))*frac_rem_cell_area)
             ELSE
c equivalent depth of water in reservoir above reservoir bottom is
c LESS than the eqivalent available storage in area outside reservoir
c below reservoir bottom
                equil_stg = (amax1(stage_res(idnstrm(ia,i),ir)
     &          - ells_in_res(idnstrm(ia,i),ir),0.0)*(sfactor(ia)
     &          /s(idnstrm(ia,i))) + amin1(ells_in_res(idnstrm(ia,i)
     &          ,ir),stage_res(idnstrm(ia,i),ir))*sfactor(ia)
     &          + h(idnstrm(ia,i))*frac_rem_cell_area)
```

```
     &          / (frac_rem_cell_area+sfactor(ia))
             ENDIF
            ENDIF
       ELSE
c
c land surface elevation in reservoir GREATER THAN  mean
c surface elevation within grid cell outside seservoir
c
c equivalent depth of water in reservoir above reservoir bottom
c elevation
          rsdpth1 = amax1(stage_res(idnstrm(ia,i),ir)
     &    - ells_in_res(idnstrm(ia,i),ir),0.0)*sfactor(ia)
c
c equivalent depth of available storage below reservoir bottom
c in grid cell outside reservoir
c
          dpth_to_wt_cell = (ells_in_res(idnstrm(ia,i),ir)
     &    - amax1(ells(idnstrm(ia,i)),stage_cell) +
     &    amax1(ells(idnstrm(ia,i)) - amax1(ells(idnstrm(ia,i))
     &    ,stage_cell),0.0) + amax1(ells(idnstrm(ia,i))
     &    - h(idnstrm(ia,i)),0.0)*s(idnstrm(ia,i)))
     &    * frac_rem_cell_area
          IF (rsdpth1 .GE. dpth_to_wt_cell) THEN
c equivalent depth of water in reservoir above reservoir bottom is
c GREATER than the eqivalent available storage in area outside reservoir
c below reservoir bottom elevation
             equil_stg = (stage_res(idnstrm(ia,i),ir)*sfactor(ia)
     &       + amax1(stage_cell,ells_in_res(idnstrm(ia,i),ir))
     &       * frac_rem_cell_area) / (frac_rem_cell_area
     &       + sfactor(ia))
          ELSE
c equivalent depth of water in reservoir above reservoir bottom is
c LESS than the eqivalent available storage in area outside reservoir
c below reservoir bottom elevation
c
c MUST recompute equivalent depth of storage in reservoir above
c the mean land surface elevation outside reservoir since we know
c the eqilibrium (mean) stage for reservoir-grid cell system is
c going to be less than the reservoir bottom elevation
             rsdpth1 = (amax1(stage_res(idnstrm(ia,i),ir)
     &       - ells_in_res(idnstrm(ia,i),ir),0.0) +
     &       amin1(stage_res(idnstrm(ia,i),ir),ells_in_res
     &       (idnstrm(ia,i),ir)) - ells(idnstrm(ia,i)) *
     &       s(idnstrm(ia,i))) * sfactor(ia)
             dpth_to_wt_cell = ((ells(idnstrm(ia,i))-h(idnstrm(ia,i)))
     &       * s(idnstrm(ia,i))- pond(idnstrm(ia,i)))*frac_rem_cell_area
             IF (rsdpth1 .GE. dpth_to_wt_cell) THEN
c equivalent depth of water in reservoir above mean cell land surface elevation
c GREATER than the eqivalent available storage in area outside reservoir
c
                equil_stg = (amax1(stage_res(idnstrm(ia,i),ir)
     &          - ells_in_res(idnstrm(ia,i),ir),0.0)* sfactor(ia)
     &          + amin1(stage_res(idnstrm(ia,i),ir),ells_in_res
     &          (idnstrm(ia,i),ir)) * s(idnstrm(ia,i)) * sfactor(ia)
     &          + amax1(stage_cell,ells(idnstrm(ia,i))) *
     &          frac_rem_cell_area) / (s(idnstrm(ia,i))
     &          * sfactor(ia) + frac_rem_cell_area)
```

```
             ELSE
c equivalent depth of water in reservoir above mean cell land surface elevation
c LESS than the eqivalent available storage in area outside reservoir
                equil_stg = (amax1(stage_res(idnstrm(ia,i),ir)
     &          - ells_in_res(idnstrm(ia,i),ir),0.0)* (sfactor(ia)
     &          / s(idnstrm(ia,i))) + amin1(stage_res(idnstrm(ia,i),ir)
     &          ,ells_in_res(idnstrm(ia,i),ir)) * sfactor(ia) +
     &          h(idnstrm(ia,i))*frac_rem_cell_area)
     &          / (sfactor(ia) + frac_rem_cell_area)
             ENDIF
          ENDIF
       ENDIF
       RETURN
       END
```

reserv_input_data.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: reserv_input_data.F,v 1.18 2005/06/10 13:29:17 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/reserv_input_data.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE READS DATA FOR RESERVOIRS/STAs
C   FILES OPENED:
C    reservour_input.dat   INPUT DEFINITION DATA FILE FOR RESERVOIRS/STAs
C    reserv_grid_loc.dat   INPUT DATA FOR GRID LOCATIONS OF RESERVOIRS
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE reserv_input_data (ires_no, ieaa_res_asr_name,
     & ieaa_res_asr_reg_name, name_res_for_inflow,
     & no_of_res_reg_frm_lok, import_area_name, ntotal_no_import_areas,
     & n_add_lev_sys, no_of_small_res_budg_files, iunit_no_res_file,
     & res_file_name, nrgtrg, noder_trg, stgr_trg, aqperm_in_res,
     & aqdep_in_res, rcpfacteaa,iopt_env_eaar_thru_g372)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'resws.inc'
```

```
      INCLUDE 'reuse.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'stormd.inc'
      INCLUDE 'canalloc.inc'

      DIMENSION aqdep_in_res(max_ncells,max_n_res_in_cell),
     & aqdep_in_res_t(max_n_res,max_n_res_grid_loc),
     & aqperm_in_res(max_ncells,max_n_res_in_cell),
     & aqperm_in_res_t(max_n_res,max_n_res_grid_loc),
     & ells_in_res_t(max_n_res,max_n_res_grid_loc),
     & icol_res_loc(max_n_res_grid_loc), ires_no(max_n_res),
     & irow_res_loc(max_n_res_grid_loc),
     & iunit_no_res_file(max_n_sm_res),
     & ixr_rtrg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & iyr_rtrg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & no_of_res_reg_frm_lok(max_n_eaa_condt),
     & noder_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & nrgtrg(max_n_res,max_nostr_res), rcpfacteaa(max_n_eaa_condt),
     & stgr_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & ixtrg_cnl(max_n_inf_loc_str),IXRLOC(max_n_out_loc_str),
     & iytrg_cnl(max_n_inf_loc_str),IYRLOC(max_n_out_loc_str)

      CHARACTER cincnl(max_n_inf_loc_str)*5, cindxaws(max_n_res)*6,
     & circnl(max_n_out_loc_str)*5, coutcnl*5,
     & ds_res_name(max_nostr_res)*6, i_ds_target_name(max_nostr_res)*5,
     & ieaa_res_asr_name(max_n_eaa_condt)*6,
     & ieaa_res_asr_reg_name(max_n_eaa_condt,max_n_reg_rel_basin)*6,
     & import_area_name(max_n_targ_area)*5,
     & name_istarf(max_n_stas_ecp)*6,CTRGCNL(max_n_out_loc_str)*5,
     & name_res_for_inflow(max_n_div_to_ecp)*6,
     & res_file_name(max_n_sm_res)*80, sim_res_name(max_n_res)*6

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  INPUT DATA FOR SIMULATED RESERVOIRS/STAs

C  initialize nunmber of reservoirs in each cell

      do node = 1,isum(maxy+1)
         noresincell(node) = 0
      enddo

C  INPUT GRID LOCATIONS FOR RESERVOIRS AND OTHER LEVEED SYSTEMS

      read (145,*) NMAREA,NRESLEC,sfactmin
      NTOTRES = NMAREA + NRESLEC
      read (145,*) no_of_small_res_budg_files
      DO i = 1,no_of_small_res_budg_files
        read (145,'(I3,2x,A6,2x,A80)') iunit_no_res_file(i)
     $  ,res_name_budg(i),res_file_name(i)
      ENDDO
      do i = 1,ntotres
         read (145,8) resname(i),nnodes(i),ibsn_no_res(i),
     $   ires_small_sim(i),ires_lev_seep_dir_cnl_opt(i),
     $   frac_seep_dir_cnl(i),frac_seep_max_lim(i),width_of_res(i),
     $   rinit_res_stg(i),(icol_res_loc(k),irow_res_loc(k),idirect(i,k),
     $   lutyp_res_loc(i,k),ells_in_res_t(i,k),aqperm_in_res_t(i,k),
```

```
     $   AQDEP_in_res_t(i,k),k = 1,nnodes(i))
         do k = 1,nnodes(i)
            node_reserv(i,k) = icol_res_loc(k) - minx(irow_res_loc(k))
     $      + 1 + isum(irow_res_loc(k))
            noresincell(node_reserv(i,k))=noresincell(node_reserv(i,k))
     $      + 1
            if (noresincell(node_reserv(i,k)) .GT. 3) THEN
               WRITE (*,3) irow_res_loc(k),icol_res_loc(k)
  3            FORMAT(/,'HAVE MORE THAN 3 RESERVOIRS IN ONE GRID CELL'
     $         ,2x,'AT COLUMN',2x,I3,2x,'AT ROW',2x,I3)
               STOP
            endif
            ires_index_in_cell(node_reserv(i,k),noresincell(node_reserv
     $      (i,k))) = i
            ells_in_res(node_reserv(i,k),noresincell(node_reserv
     $      (i,k))) = ells_in_res_t(i,k)
            lutyp_in_res(node_reserv(i,k),noresincell(node_reserv
     $      (i,k))) = lutyp_res_loc(i,k)
            IF (aqperm_in_res_t(i,k) .gt. 0.) then
              aqperm_in_res(node_reserv(i,k),noresincell(node_reserv
     $        (i,k))) = aqperm_in_res_t(i,k)*10000.
            ELSE
              aqperm_in_res(node_reserv(i,k),noresincell(node_reserv
     $        (i,k))) = tkx(node_reserv(i,k))
            ENDIF
            IF (AQDEP_in_res_t(i,k) .gt. 0.) then
              AQDEP_in_res(node_reserv(i,k),noresincell(node_reserv
     $        (i,k))) = AQDEP_in_res_t(i,k)
            ELSE
              AQDEP_in_res(node_reserv(i,k),noresincell(node_reserv
     $        (i,k))) = aqdep(node_reserv(i,k))
            ENDIF
         enddo
      enddo
      read (145,*) resname(ntotres+1),nnodes(ntotres+1)
c  other leveed systems
      kk = 2
  5   read (145,9,end = 7) resname(ntotres+kk),nnodes(ntotres+kk)
     $   ,ibsn_no_res(ntotres+kk),(icol_res_loc(k),irow_res_loc(k)
     $   ,lutyp_res_loc(ntotres+kk,k),k = 1,nnodes(ntotres+kk))
      do k = 1,nnodes(ntotres+kk)
         node_reserv(ntotres+kk,k) = icol_res_loc(k)
     $   - minx(irow_res_loc(k)) + isum(irow_res_loc(k))
      enddo
  8   format(A6,2x,2i3,2(2x,A3),1x,3F5.1,F6.1,1x,20(1x,4i3,3F5.1))
  9   format(A6,2x,2i3,1x,20(1x,3i3))
      kk = kk + 1
      go to 5
c
C--------------------------------
c  INPUT DATA FOR SIMULATED RESERVOIRS/STAs
c--------------------------------
c
  7   n_add_lev_sys = kk - 1
      IF (NTOTRES .GT. 0) THEN
c
         DO i = 1,neaabsn
```

```
            CALL match(ieaa_res_asr_name(i),1,6,resname,ntotres+1,6,
     $      imatch)
            IF (imatch .eq. 0) THEN
               WRITE(*,100) ieaa_res_asr_name(i),i
               STOP
            ENDIF
            ieaa_res_asr(i) = imatch
         END DO
         DO i = 1,neaabsn
            DO k = 1,no_of_res_reg_frm_lok(i)
               CALL match(ieaa_res_asr_reg_name(i,k),1,6,resname,ntotres
     $         +1, 6,imatch)
               IF (imatch .eq. 0) THEN
                  WRITE(*,200) ieaa_res_asr_reg_name(i,k),i
                  STOP
               ENDIF
               ieaa_res_asr_reg(i,k) = imatch
            END DO
         END DO
      ENDIF
      DO i = 1,no_of_reuse_plnts
         DO k=1,no_grid_cells_reuse(i)
            CALL match(resname_reuse(i,k),1,6,resname,ntotres + 1,6,
     $      imatch)
            IF (imatch .eq. 0) THEN
               WRITE(*,300) resname_reuse(i,k),i
               STOP
            ENDIF
            ires_no_reuse(i,k) = imatch
         END DO
      END DO
      IF (NBSNTSTA .GT. 0) THEN
         DO i = 1,NBSNTSTA
            CALL match(name_res_for_inflow(i),1,6,resname,ntotres + 1,6
     $      ,imatch)
            IF (imatch .eq. 0) THEN
               WRITE(*,400) name_res_for_inflow(i),i
               STOP
            ENDIF
            indxsta(i) = imatch
         END DO
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      indx_sta1e_frm_lok_ws = ntotres+1
      indx_sta1w_frm_lok_ws = ntotres+1
      indx_sta2w_frm_lok_ws = ntotres+1
      indx_sta1ee_frm_lok_ws = ntotres+1
c
      READ (103,'(i5,2x,10(A6,1x))') NMAWS,(CINDXAWS(I),I=1,NMAWS)
      DO i = 1,NMAWS
         CALL match(CINDXAWS(I),1,6,resname,ntotres + 1,6,imatch)
         IF (imatch .eq. 0) THEN
            WRITE(*,500) CINDXAWS(I)
            STOP
         ENDIF
         INDXAWS(I) = imatch
      END DO
```

```
c
c
      IF (NMAWS .GT. 0) READ (103,80) (IDIRFRLO(INDXAWS(I)),I=1,NMAWS)
c
      READ (103,'(I5,2x,10(A6,1x))')NSTARFF,(name_ISTARF(I),I=1,NSTARFF)
      DO i = 1,NSTARFF
         CALL match(name_ISTARF(i),1,6,resname,ntotres + 1,6,imatch)
         istarf(i) = imatch
      END DO
      READ (103,*) NPUMPTORES,(IBSNRES(I),PUMPCAP_RES(I) ,I = 1,
     $NPUMPTORES)
      READ (103,*) iopt_env_eaar_thru_g372
      READ (103,'(i3,2x,25(A6,1X))') no_of_res_spec_code ,
     $(name_res_spec_code(i),i=1,no_of_res_spec_code)
      READ (103,'(I3,2x,50(A6,1x))') no_of_struc_spec_code ,
     $(res_struc_name_spec_code(i),i=1,no_of_struc_spec_code)
c
c
      IF (NTOTRES .GT. 0) THEN
c
         DO 60 IR = 1,NTOTRES
c
c
            READ (103,'(A6)') sim_res_name(ir)
            print *,sim_res_name(ir),'  RES'
            CALL match (sim_res_name(ir),1,6,resname,ntotres+1,6,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,10) sim_res_name(ir)
               STOP
            ENDIF
            ia = imatch
            READ (103,*) RES_AREA(IA)
            READ (103,*) STGMX(IA)
c
            STGMX2(IA) = STGMX(IA)
c
            READ (103,*) DPHWS(IA)
            READ (103,*) IXCNODE(IA),IYCNODE(IA)
c
            READ (103,*) NINSTR(IA)
            IF (NINSTR(IA) .GT. 0) THEN
               READ (103,'(10(A6,1x))') (res_struc_inflow_name(ia,i) ,i
     $         = 1,ninstr(ia))
               READ (103,70) (WEIRCFF(IA,I,1),I = 1,NINSTR(IA))
               READ (103,70) (CRSTELEV(IA,I,1),I = 1,NINSTR(IA))
               READ (103,70) (WREXP(IA,I,1),I = 1,NINSTR(IA))
               IF (resname(ia) .eq. 'STA34 ') THEN
                 DO i = 1,ninstr(ia)
                   IF (res_struc_inflow_name(ia,i) .EQ. 'G372  ') THEN
                     g372cap = WEIRCFF(IA,I,1)
                   ELSEIF (res_struc_inflow_name(ia,i) .EQ. 'G370  ')
     $             THEN
                     g370cap = WEIRCFF(IA,I,1)
                   ENDIF
                 ENDDO
               ENDIF
C
```

```
            DO IS = 1,NINSTR(IA)
               READ (103,80) NINDPTS(IA,IS),INODOPT(IA,IS,1)
               IF (INODOPT(IA,IS,1) .EQ. 1) THEN
                  READ (103,80) (IXLIN(IA,IS,I),IYLIN(IA,IS,I),I = 1,
     $            NINDPTS (IA,IS))
               ELSE
                  READ (103,'(10(A5,1X))') (CINCNL(I),I=1,NINDPTS
     $            (IA,IS))
                  DO I = 1,NINDPTS(IA,IS)
                     CALL match(CINCNL(I),1,5,cnm,nch,5,imatch)
                     IF (imatch .EQ. 0) THEN
                        WRITE(*,20) CINCNL(I),RESNAME(IA)
                        STOP
                     ENDIF
                     INCNL(IA,IS,I) = imatch
                  END DO
               ENDIF
            END DO
         ENDIF
c
C
         READ (103,*) noutstr(ia)
         noutws = 0
c
         IF (NOUTSTR(IA) .GT. 0) THEN
            READ (103,'(10(A6,1x))') (res_out_strname(ia,i),i = 1,
     $      noutstr(ia))
            IF (resname(ia) .EQ. name_res_spec_code(1)) THEN
               no_gravity_outlets_sta34 = 0
               DO i = 1,noutstr(ia)
                  IF (res_out_strname(ia,i) .EQ.
     $            res_struc_name_spec_code(1) .OR. res_out_strname
     $            (ia,i) .EQ. res_struc_name_spec_code(2) .OR.
     $            res_out_strname(ia,i) .EQ. res_struc_name_spec_code
     $            (3)) THEN
                     no_gravity_outlets_sta34 =
     $               no_gravity_outlets_sta34 + 1
                     igrav_outlet_no(no_gravity_outlets_sta34) = i
                  ENDIF
               END DO
            ENDIF
            READ (103,'(10(A3,3x))') (opt_sim_code(ia,i),i=1,noutstr
     $      (ia))
            READ (103,70) (WEIRCFF(IA,I,2),I=1,NOUTSTR(IA))
            READ (103,70) (CRSTELEV(IA,I,2),I=1,NOUTSTR(IA))
            DO I=1,NOUTSTR(IA)
               CRSTELEV2(IA,I) = CRSTELEV(IA,I,2)
            ENDDO
            READ (103,70) (WREXP(IA,I,2),I=1,NOUTSTR(IA))
            READ (103,70) (RMXCPO(IA,I),I=1,NOUTSTR(IA))
            READ (103,'(10(A6,1x))') (ds_res_name(i),I=1,NOUTSTR(IA))
            DO i = 1,noutstr(ia)
               CALL match(ds_res_name(i),1,6,resname,ntotres+1,6,
     $         imatch)
               IF (imatch .NE. 0) THEN
                  iwmads(ia,i) = imatch
               ELSE
```

```
                  iwmads(IA,I) = ntotres+1
               ENDIF
            END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            READ (103,'(10(A7,1X))') (TYPE(IA,I),I=1,NOUTSTR(IA))
            READ (103,'(10A6)') (res_out_type(ia,i),i =1,noutstr(ia))
            READ (103,*) (i_ds_target_name(i),i = 1,noutstr(ia))
c
            DO i=1,noutstr(ia)
               CALL match (i_ds_target_name(i),1,5,import_area_name ,
     $         ntotal_no_import_areas+1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,30) i_ds_target_name(i),resname(ia)
                  STOP
               ENDIF
               i_ds_target_indx(ia,i) = imatch
            END DO
c
            DO i = 1,noutstr(ia)
               IF (res_out_type(ia,i) .EQ. 'WSPLY' .OR. res_out_type
     $         (ia,i) .EQ. 'SEEPG') THEN
                  noutws = noutws + 1
                  istrctr_ws(ia,noutws) = i
               ENDIF
            END DO
            IF (noutws .GT. 0) THEN
               nres_ws = nres_ws + 1
               ires_no(nres_ws) = ia
               noutlets(nres_ws) = noutws
               READ (103,*) iconv_use_opt_ws(nres_ws)
               DO is = 1,noutws
                  READ (103,*) idshg_opt(ires_no(nres_ws),is)
                  READ (103,*) ieaa_bsn(nres_ws,is)
                  READ (103,*) stg_min_wsply(ires_no(nres_ws),is)
               END DO
            ENDIF
         ENDIF
c
c  More Detailed Information for Outflow structure(s)
c
         IF (NOUTSTR(IA) .GT. 0) THEN
            DO IS = 1,NOUTSTR(IA)
               READ (103,80) NOUTDPTS(IA,IS),INODOPT(IA,IS,2)
               IF (INODOPT(IA,IS,2) .EQ. 1) THEN
                  READ (103,80) IXLOUT(IA,IS),IYLOUT(IA,IS)
               ELSE
                  READ (103,'(a5)') COUTCNL
                  CALL match(COUTCNL,1,5,cnm,nch + 1,5,imatch)
                  IF (imatch .EQ. 0) THEN
                     WRITE(*,40) COUTCNL,RESNAME(IA)
                     STOP
                  ENDIF
                  IOUTCNL(IA,IS) = imatch
               ENDIF
               READ (103,80)NDSPTS(IA,IS),INODOPT(IA,IS,3)
               IF (INODOPT(IA,IS,3) .EQ. 1) THEN
                  READ (103,*) (IXRW(IA,IS,I),IYRW(IA,IS,I) ,I=1,
```

```
     $                NDSPTS(IA,IS)),NRGTRG(IA,IS),(IXR_RTRG(IA,IS,J)
     $                ,IYR_RTRG(IA,IS,J),STGR_TRG(IA,IS,J),J=1
     $                ,NRGTRG(IA,IS))
                       DO I=1,NRGTRG(IA,IS)
                         NODER_TRG(IA,IS,I)=IXR_RTRG(IA,IS,I)
     $                   -MINX(IYR_RTRG(IA,IS,I))+1
     $                   +ISUM(IYR_RTRG(IA,IS,I))
                       ENDDO
                  ELSE
                     READ (103,*) (CIRCNL(I),IXRLOC(I),IYRLOC(I),
     $               CTRGCNL(I),IXTRG_CNL(I),IYTRG_CNL(I),DWNSTGMX
     $               (IA,IS ,I),OFFSET_REG_RES(IA,IS,I),I=1,NDSPTS(IA
     $               ,IS))
                     DO I=1,NDSPTS(IA,IS)
                        CALL match (CIRCNL(I),1,5,cnm,nch + 1,5,imatch)
                        IF (imatch .EQ. 0) THEN
                           WRITE(*,50) CIRCNL(I),RESNAME(IA)
                           STOP
                        ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                        IRCNL(IA,IS,I) = imatch
                        CALL match (CTRGCNL(I),1,5,cnm,nch + 1,5,imatch)
                        IF (imatch .EQ. 0) THEN
                           WRITE(*,55) CTRGCNL(I),RESNAME(IA)
                           STOP
                        ENDIF
                        ITRGCNL(IA,IS,I) = imatch
                        ipos_match = 0
                        DO IC = 1,NODCR(ITRGCNL(IA,IS,I))
                           IF (IXTRG_CNL(I).EQ. XCN(ITRGCNL(IA,IS,I),IC)
     $                     .AND. IYTRG_CNL(I) .EQ. YCN(ITRGCNL(IA,IS,I)
     $                     ,IC)) THEN
                              NODEPOS_RCNL_TRG(IA,IS,I) = IC
                              ipos_match = 1
                           ENDIF
                        ENDDO
                        ipos_matchr = 0
                        DO IC = 1,NODCR(IRCNL(IA,IS,I))
                           IF (IXRLOC(I).eq.XCN(IRCNL(IA,IS,I),IC)
     $                     .AND. IYRLOC(I) .eq.YCN(IRCNL(IA,IS,I),IC))
     $                     THEN
                             NODEPOS_CANLR(IA,IS,I) = IC
                             ipos_matchr = 1
                           ENDIF
                        ENDDO
                        if (IXTRG_CNL(I) .GT. 0 .and. IYTRG_CNL(I)
     $                   .GT. 0) THEN
                           icell_trg_loc(ia,is,i) = IXTRG_CNL(I)
     $                     - MINX(IYTRG_CNL(I)) + 1 + ISUM(IYTRG_CNL(I))
                        ELSE
                           icell_trg_loc(ia,is,i) = -901
                        ENDIF
                        IF (ipos_matchr .eq. 0 .and. CIRCNL(I) .ne.
     $                   'NOCNL') THEN
                            WRITE(*,58) IXRLOC(I),IYRLOC(I),CIRCNL(I)
                            STOP
                        ENDIF
```

```
                        IF (ipos_match .eq. 0 .and. CTRGCNL(I) .ne.
     $                   'NOCNL') THEN
                            WRITE(*,57) IXTRG_CNL(I),IYTRG_CNL(I),
     $                      CTRGCNL(I)
                            STOP
                        ENDIF
                     END DO
                  ENDIF
               END DO
            ENDIF
   60    CONTINUE
      ENDIF
C
      RETURN
   10 format(2(/),'UNRECOGNIZED RESERVIOR NAME',2x,A6,2x,'CHECK RESERVO
     $IR INPUT FILE')
   20 format(2(/),'UNRECOGNIZED INLET CANAL NAME',2x,A5,2x,
     $'FOR  RESERVOIR',2x,A5,2x,'CHECK RESERVOIR INPUT FILE')
   30 format(2(/),'UNRECOGNIZED DS TARGET NAME',2x,A5,2X,
     $'FOR RESERVOIR',2x,A6,2x,'CHECK RESERVOIR INPUT FILE')
   40 format(2(/),'UNRECOGNIZED OUTLET CANAL NAME',2x,A5,2x,
     $'FOR  RESERVOIR',2x,A6,2x,'CHECK RESERVOIR INPUT FILE')
   50 format(2(/),'UNRECOGNIZED DS CANAL NAME',2x,A5,'FOR RESERVOIR',2x,
     $A6,2x,'CHECK RESERVOIR INPUT FILE')
   55 format(2(/),'UNRECOGNIZED CANAL NAME',2x,A5,' FOR TRIGGERING DISCH
     $ARGES FROM RESERVOIR',2x,A6,2x,'CHECK RESERVOIR INPUT FILE')
   57 format(2(/),'UNRECOGNIZED GRID LOCATION',2x,2i5,2x,'FOR TRIGGER CA
     $NAL',2x,A5,2x,'CHECK RESERVOIR INPUT FILE')
   58 format(2(/),'UNRECOGNIZED GRID LOCATION',2x,2i5,2x,'FOR DOWNSTREAM
     $ CANAL',2x,A5,2x,'CHECK RESERVOIR INPUT FILE')
   70 FORMAT (30F6.0)
   80 FORMAT(50I6)
  100 FORMAT(2(/),'UNRECOGNIZED EAA RESERVOIR',2x,A6,2x,'STORING EAA RUN
     $OFF FOR EAA BASIN NO',2x,I3,2x,'CHECK MODEL_DEFINITION_INFO.DAT FI
     $LE')
  200 FORMAT(2(/),'UNRECOGNIZED EAA RESERVOIR',2x,A6,2x,'STORING EXCESS
     $LOK WATER FOR EAA BASIN NO',2x,I3,2x,'CHECK MODEL_DEFINITION_INFO.
     $DAT FILE')
  300 FORMAT(2(/),'UNRECOGNIZED RESERVOIR',2x,A6,2x,'NAME STORING REUSE
     $WATER',2x,'FOR REUSE PLANT NO',2x,I2,2x,'CHECK MODEL_DEFINITION_IN
     $FO.DAT FILE')
  400 FORMAT(2(/),'UNRECOGNIZED STA FOR ECP',2x,A6,2x,'TREATING WATER FR
     $OM BASIN NO',2x,I2,2x,'INPUT. CHECK MODEL_DEFINITION_INFO.DAT FILE
     $')
  500 FORMAT(2(/),'UNRECOGNIZED RESERVOIR NAME',2x,A6,2x,'FOR RESERVOIR
     $ THAT IS MAINTAINED AT SOME MINIMUM DEPTH. CHECK MODEL_DEFINITION_
     $INFO.DAT FILE')
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

reservoir_init.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: reservoir_init.F,v 1.9 2005/06/10 13:29:18 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/reservoir_init.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine initializes parameters necessary for
C   simulation of reservoirs.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE reservoir_init (volwtr, sumrain, sumrain_unadj, jday,
     & gcell_mean_ponding_depth, gcell_mean_ponding_depth_end, rain,
     & prev_stage_res)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wrout.inc'

      DIMENSION avg_chg_depth_levee_spg(max_n_res),
     & avg_et_res(max_n_res), avg_rain_res(max_n_res),
     & avg_seep_loss_res(max_n_res), cum_loss_res(max_n_res),
     & gcell_mean_ponding_depth(max_ncells),
     & gcell_mean_ponding_depth_end(max_ncells),
     & prev_stage_res(max_ncells,max_n_res_in_cell), rain(max_ncells),
     & sumrain(max_ncells), sumrain_unadj(max_ncells), volwtr(max_n_res)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      DO IA = 1,NTOTRES
         IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
```

```
     &   (ia) .eq. 'NO ') THEN
         rmean_dpth_rel_to_ls = 0.0
         imaint = 0
         DO jj = 1,NMAWS
            IF (ia .EQ. indxaws(jj)) THEN
               imaint = imaint + 1
            ENDIF
         END DO
         DO  IY = MAXY,1,-1
            DO IX = MINX(IY),MAXX(IY)
            NODE = IX - MINX(IY) + 1 + ISUM(IY)
            stage_cell = h(node) + pond(node)
            DO K = 1,NNODES(ia)
            IF (NODE .EQ. node_reserv(ia,k)) THEN
               depth_of_deficit = SUMET_outside_res(node) +
     $         seep_loss_res_cell(node) - SUMRAIN_unadj(node) / 12.0 -
     $         sum_chg_depth_levee_spg_unadj(NODE) -
     $         (outf_fc_frm_res_to_cell(NODE) + sum_chg_dpth_ovl_flw
     $         (NODE))
               depth_of_deficit = amin1(depth_of_deficit,0.0)
               call etcomp(1,et,node,lutyp(node),jday,depth_of_deficit ,
     $         stage_res,ET_outside_res,ia,1,0.0,ells_in_res,0)
               volwtr(ia)= volwtr(ia) + et * gdar * sfactor(ia) / 86400.
               rmean_dpth_rel_to_ls = amax1(stage_cell - ells(node),0.0)
     $         - amax1(ells(node) - stage_cell,0.0) * s(node)
     $         + rmean_dpth_rel_to_ls
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            ENDIF
            END DO
            END DO
         END DO
      IF (imaint .GT. 0 .AND.rmean_dpth_rel_to_ls .GT. 0.) THEN

         tot_rain = 0.0
         tot_et = 0.0
         nnode_res = 0
         tot_levee_spg_loss = 0.0
         tot_seep_loss_res = 0.0
         node_rep_res = ixcnode(ia) - minx(iycnode(ia)) +  1+ isum
     $   (iycnode(ia))
         rep_stage_res = h(node_rep_res) + pond(node_rep_res)
         IF (rep_stage_res .LE. ells(node_rep_res) + DPHWS(IA)*sfactor
     $   (ia) .OR. stage_res(node_rep_res,1) .LE. ells(node_rep_res) +
     $   DPHWS(IA)) THEN
            DO  IY = MAXY,1,-1
               DO IX = MINX(IY),MAXX(IY)
               NODE = IX - MINX(IY) +  1+ ISUM(IY)
               DO K = 1,NNODES(ia)
               IF (NODE .EQ. node_reserv(ia,k)) THEN
                  depth_of_deficit = SUMET_outside_res(node) +
     $            seep_loss_res_cell(NODE) - SUMRAIN_unadj(node)/12.0 -
     $            sum_chg_depth_levee_spg_unadj(NODE) -
     $            (outf_fc_frm_res_to_cell(NODE) + sum_chg_dpth_ovl_flw
     $            (NODE))
                  depth_of_deficit = amin1(depth_of_deficit,0.0)
                  call etcomp(1,et,node,lutyp(node),jday,
     $            depth_of_deficit,stage_res,ET_outside_res,ia,1,0.0
```

```
     $            ,ells_in_res,0)
                  tot_rain = tot_rain + rain(node)/12.
                  tot_et = tot_et + et
                  nnode_res = nnode_res + 1
                  tot_levee_spg_loss = tot_levee_spg_loss +
     $            chg_depth_levee_spg(node)
                  tot_seep_loss_res = tot_seep_loss_res +
     $            gcell_mean_ponding_depth(node) -
     $            gcell_mean_ponding_depth_END(node)
               ENDIF
               END DO
               END DO
            END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            avg_rain_res(ia) = tot_rain/nnode_res + avg_rain_res(ia)
            avg_et_res(ia) = tot_et/nnode_res + avg_et_res(ia)
            avg_seep_loss_res(ia) = tot_seep_loss_res/nnodes(ia) +
     $      avg_seep_loss_res(ia)
            avg_chg_depth_levee_spg(ia) = tot_levee_spg_loss/nnodes(ia)+
     $      avg_chg_depth_levee_spg(ia)
            cum_loss_res(ia) = avg_et_res(ia) + avg_seep_loss_res(ia) -
     $      avg_chg_depth_levee_spg(ia) - avg_rain_res(ia)
         ELSE
            avg_rain_res(ia) = 0.0
            avg_et_res(ia) = 0.0
            avg_seep_loss_res(ia) = 0.0
            avg_chg_depth_levee_spg(ia) = 0.0
            cum_loss_res(ia) = 0.0
         ENDIF
      ELSE
      avg_rain_res(ia) = 0.0
      avg_et_res(ia) = 0.0
      avg_seep_loss_res(ia) = 0.0
      avg_chg_depth_levee_spg(ia) = 0.0
      cum_loss_res(ia) = 0.0
      ENDIF
      IF (rmean_dpth_rel_to_ls .LE. 0. .OR. (imaint .GT. 0 .AND.
     $cum_loss_res(ia) .GE. DPHWS(ia)*sfactor(ia))) THEN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      DO IY = MAXY,1,-1
         DO IX = MINX(IY),MAXX(IY)
         NODE = IX - MINX(IY) +  1+ ISUM(IY)
         DO k = 1,nnodes(ia)
         IF (NODE .EQ. node_reserv(ia,k)) THEN
            SUMRAIN(NODE) = 0.0
            SUMRAIN_unadj(NODE) = 0.0
            SUMET(NODE) = 0.0
            SUMET_outside_res(NODE) = 0.0
            adjust_dur_excess(node) = 0.0
            adjust_dur_deficit(node) = 0.0
            adjust_dur_deficit0(node) = 0.0
            prev_adjust_for_area(node) = 0.0
            adjust_for_area(NODE) = 0.0
            adjust_for_area_n(NODE) = 0.0
            adjust_for_area_m(NODE) = 0.0
            adjust_for_area_m1(NODE) = 0.0
            adjust_for_area_m2(NODE) = 0.0
```

```
            adjust_for_area_m3(NODE) = 0.0
            adjust_for_area_m4(NODE) = 0.0
            sum_chg_depth_levee_spg(NODE) = 0.0
            sum_chg_depth_levee_spg_unadj(NODE) = 0.0
            sum_chg_dpth_ovl_flw(NODE) = 0.0
            outf_fc_frm_res_to_cell(NODE) = 0.0
            outf_seep_loc_to_res_cell(NODE) = 0.0
            seep_loss_res(node) = 0.0
            seep_loss_res_cell(node) = 0.0
            daily_outf_fc_frm_res_to_cell(node) = 0.0
            pond_adj_due_to_res(node) = 0.0
         ENDIF
         END DO
         END DO
      END DO
      rinflw_to_res_frm_cnl(ia) = 0.0
      ENDIF
      ENDIF
      END DO
c
      DO NODE = 1,ISUM(MAXY)
        DO ir = 1,noresincell(NODE)
           prev_stage_res(node,ir) = stage_res(node,ir)
        ENDDO
      ENDDO
      RETURN
      END
```

resout.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: resout.F,v 1.19 2005/06/10 13:29:18 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/resout.F,v $
C   $Name: rel-5-5 $
C
C    THIS SUBROUTINE CALCULATES STRUCTURAL OUTFLOW AND
C    SEEPAGE FROM LEVEED SYSTEMS (MAINLY RESERVOIRS).
C    OUTFLOW CAN BE ROUTED TO GRID CELL(S),CANAL,OR TO
C    ANOTHER RESERVOIR.
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE resout (volwtr, sumrain, out_to_wca1_frm_sta1w,
     & out_to_wca1_frm_sta1e, outflow_to_canal_roten, outflow_to_marsh,
     & ifacteaa, convey_canal_names, rstgpls, stgref_asr, sem_dmnd_orig,
     & rloc_excss_to_canl_next_step, dischg_to_bay_s29_frm_c11r, wcaenv,
     & tot_cnl_dmd_asr, trig_wca_stg, regstg_wca, rain,
     & flow_to_cpbres_frm_e1, offset_to_sched,
     & iopt_for_semcyp_prior_roten, flow_to_c9r_frm_c9,
     & flow_to_lbn_frm_c11, tot_flw_to_c9r_frm_nlkblt, rnlkblt_to_s29,
     & flow_to_ca1_frm_acmebsn, ws_from_acmres_lwdd,total_afact,
     & total_small_afact ,nrgtrg,noder_trg,stgr_trg, tvolwtrl,
     & avvolgr_mx, tvolwtrl_p,no_canl_in_cell,icanl_no_in_cell,
     & ipos_cell_in_canl,rindiv_canl_area,total_cell_stor,
     & frac_s319_sta1ew,frac_s319_sta1ee,ratio_flow_s6_sta2e,
     & ratio_flow_s6_sta2m,ratio_flow_s6_sta2w)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
```

```
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'loktarg.inc'
      INCLUDE 'resbudg.inc'

      DIMENSION avvolgr(max_n_res), avvolgr_mx(max_n_res,max_nostr_res),
     & daily_outf_fc_frm_res_to_cell1(max_ncells),
     & noder_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & nrgtrg(max_n_res,max_nostr_res),
     & offset_to_sched(max_n_stor_w_oper_sch), rain(max_ncells),
     & regstg_wca(max_n_wcas,max_n_zones_wca),
     & rloc_excss_to_canl_next_step(max_ncnls), rstgpls(max_n_puls_rel),
     & stgr_trg(max_n_res,max_nostr_res,max_n_cells_constrt),
     & sumrain(max_ncells), total_afact(max_ncells),
     & total_small_afact(max_ncells), trig_wca_stg(max_n_wcas),
     & tvolwtrl(max_n_res), tvolwtrl_p(max_n_res),
     & vol_to_asr_frm_res(max_src_dest_pts_for_any_asr),
     & vol_to_res_frm_asr(max_src_dest_pts_for_any_asr),
     & volwtr(max_n_res),no_canl_in_cell(max_ncells),
     & icanl_no_in_cell(max_ncells,3),rindiv_canl_area(max_ncells,3),
     & ipos_cell_in_canl(max_ncells,3),canl_stor_in_cell(max_ncells),
     & cell_storage(max_ncells),total_cell_stor(max_ncells)

      CHARACTER convey_canal_names(max_n_tot_conv_cnl_wcas)*5,
     & wcaenv(max_n_wcas)*5

      DOUBLE PRECISION roten_outflw_stage,crest_elev_struc
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      total_outflow_from_roten=0.0
      outflow_from_roten_to_west=0.0
      pot_exc_outf_frm_sta34_to_nca3a=0.0
      dischg_to_s7_frm_sta34=0.0
      dischg_to_s8_frm_sta34=0.0
      sta34_out_to_wca3ane=0.0
      roten_stage=(h(node_roten1)+pond(node_roten1)+h(node_roten2)+
     &pond(node_roten2))/2.0
      IF (igravity_s8.eq.1) THEN
        s8capac=s8_cap
      ELSE
        s8capac=s8max
      END IF
      IF (igravity_s7.eq.1) THEN
        s7capac=s7_cap
      ELSE
        s7capac=s7max
      END IF
      ref_stg_for_cell_stor=-100.
```

```
      DO node = 1,max_ncells
        canl_stor_in_cell(node) = 0.0
        cell_storage(node) = 0.0
      ENDDO
!
!
      DO ia=1,ntotres
         tvolwtrl(ia)=volwtr(ia)
         tvolwtrl_p(ia)=volwtr(ia)
      ENDDO
      DO 50 ia=1,ntotres
        total_outflow_from_sta=0.0
!
!  calculate adjustment in estimate of reservoir stage due to
!   differences in actual reservoir area and total area of grid
!   cells in which reservoir is located
!
        IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &   (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
        DO node=1,isum(maxy+1)
        DO k = 1,noresincell(node)
          IF (ia .eq. ires_index_in_cell(node,k)) THEN
            stage=h(node)+pond(node)
            IF (stage.le.ells(node)) THEN
              afact=1.0
            ELSE
              afact=total_afact(node)
            END IF
            adjust_for_ovlf=sum_chg_dpth_ovl_flw(node)
            term1_for_adjust_area=(sumet_outside_res(node)
     &       +seep_loss_res(node)-sumrain(node)/12.0
     &       -sum_chg_depth_levee_spg(node))
     &       *((1.0-afact)/afact)
            term2_for_adjust_area=(outf_fc_frm_res_to_cell(node)+
     &       daily_outf_fc_frm_res_to_cell(node)+
     &       outf_seep_loc_to_res_cell(node)+adjust_for_ovlf)/afact
            daily_outf_fc_frm_res_to_cell1(node)=
     &       daily_outf_fc_frm_res_to_cell(node)
            IF (stage.ge.ells(node)) THEN
!
              IF (afact.le.1.0) THEN
                sgn=1
              ELSE
                sgn=-1
              END IF
              IF (sgn*
     &         term1_for_adjust_area.lt.term2_for_adjust_area.and.stage.
     &         gt.ells(node)) THEN
               adjust_dur_excess(node)=adjust_for_area_m(node)+
     &          adjust_dur_deficit(node)
               adjust_for_area_m1(node)=0.0
c              if (resname(ia).eq.'STA1W ') print *,adjust_for_area_m
c    &          (node),adjust_dur_deficit(node),adjust_dur_excess(node)
c    &           ,'   TERM1LTERM2  ',node,term1_for_adjust_area
c    &           ,term2_for_adjust_area,sumet_outside_res(node)
```

```
c    &            ,seep_loss_res(node),sumrain(node)/12.0
               ELSE
c                if (resname(ia).eq.'STA1W ') print *,
c    &           adjust_dur_deficit(node),adjust_dur_excess(node),' DFDF'
                 adjust_dur_deficit(node)=adjust_dur_excess(node)
                 adjust_dur_excess(node)=adjust_dur_deficit(node)
c                if (resname(ia).eq.'STA1W ') print *,adjust_for_area_m
c    &           (node),adjust_dur_deficit(node),adjust_dur_excess(node)
c    &           ,'  TRTRTR  ',node,term1_for_adjust_area
c    &           ,term2_for_adjust_area,sumet_outside_res(node)
c    &           ,seep_loss_res(node),sumrain(node)/12.0
               END IF
               depsw=(stage-ells(node))/afact+term1_for_adjust_area-
     &          (outf_fc_frm_res_to_cell(node)+
     &          outf_seep_loc_to_res_cell(node)+
     &          daily_outf_fc_frm_res_to_cell(node))/afact-
     &          sum_chg_dpth_ovl_flw(node)/afact-adjust_dur_excess(node)
               prev_adjust_for_area(node)=term1_for_adjust_area-
     &          term2_for_adjust_area
               IF (sgn*
     &          term1_for_adjust_area.ge.term2_for_adjust_area.or.stage
     &          .le.ells(node)) THEN
                 adjust_for_area_m(node)=term1_for_adjust_area-
     &            term2_for_adjust_area+daily_excess_adjust(node)
                 prev_adjust_for_area(node)=sgn*amin1(sgn*
     &            adjust_for_area_m(node),sgn*(depsw-(stage-ells(node))))
                 IF (afact.le.1.0) adjust_for_area_m(node)=
     &            amax1(adjust_for_area_m(node),0.0)
                 adjust_for_area_m(node)=sgn*amin1(sgn*
     &            adjust_for_area_m(node),sgn*(depsw-(stage-ells(node))))
                 IF (stage.le.ells(node)) THEN
                   IF (depsw.gt.0.000001) THEN
                     adjust_for_area_m(node)=daily_et(node)+
     &                daily_seep_loss_res(node)+chg_depth_levee_spg(node)
     &                -daily_outf_fc_frm_res_to_cell(node)-
     &                daily_outf_seep_loc_to_res_cell(node)-rain(node)/
     &                12.0
                     prev_adjust_for_area(node)=
     &                amin1(adjust_for_area_m(node),depsw)
                     adjust_for_area_m(node)=
     &                amax1(adjust_for_area_m(node),0.0)
                     adjust_for_area_m(node)=
     &                amin1(adjust_for_area_m(node),depsw)
                   END IF
                 END IF
                 depsw = depsw - adjust_for_area_m(node)
c              if (resname(ia).eq.'STA1W ') print *,term1_for_adjust_area
c    $           ,term2_for_adjust_area,adjust_for_area_m(node)
c    $           ,depsw,stage-ells(node),'  DEPSW ADJ  ',afact
c    $           ,daily_excess_adjust(node),adjust_dur_excess(node)
c    $           ,sumet_outside_res(node),seep_loss_res(node)
c    $           ,sumrain(node)/12.0,adjust_dur_deficit(node)
                 adjust_for_area_m1(node)=adjust_for_area_m(node)
               END IF
             END IF
           END IF
         END DO
```

```
        END DO
        ENDIF
!
!  calculate available storage in reservoir to be injected into asr
!
        IF (nasr_from_res(ia).gt.0) THEN
          DO ij=1,nasr_from_res(ia)
            DO k=1,nnodes_res_to_asr(ia,ij)
              node=icol_asr(ia,ij,k)-minx(irow_asr(ia,ij,k))+1+
     &         isum(irow_asr(ia,ij,k))
              DO ir = 1,noresincell(node)
              IF (ia .eq. ires_index_in_cell(node,ir)) THEN
              IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &        (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
                stage1=pond(node)-(ells(node)-h(node))*s(node)
     &          +ells(node)
                IF (stage1.le.ells(node)) THEN
                  afact=1.0
                ELSE
                  afact=total_afact(node)
                END IF
                IF (afact.le.1.0) THEN
                  sgn=1
                ELSE
                  sgn=-1
                END IF
                IF (stage1.gt.ells(node)) THEN
                   adjust_for_ovlf=sum_chg_dpth_ovl_flw(node)
                   term1_for_adjust_area=(sumet_outside_res(node)
     &             +seep_loss_res(node)-sumrain(node)/12.0
     &             -sum_chg_depth_levee_spg(node))
     &             *((1.0-afact)/afact)
                   term2_for_adjust_area=(outf_fc_frm_res_to_cell
     &             (node)+outf_seep_loc_to_res_cell(node)+
     &             daily_outf_fc_frm_res_to_cell(node)+adjust_for_ovlf)
     &             /afact
                   depsw=(stage1-ells(node))/afact
     &             +term1_for_adjust_area-(outf_fc_frm_res_to_cell
     &             (node)+outf_seep_loc_to_res_cell(node)+
     &             daily_outf_fc_frm_res_to_cell(node))/afact-
     &             sum_chg_dpth_ovl_flw(node)/afact-
     &             adjust_dur_excess(node)
                   depsw=depsw-adjust_for_area_m1(node)
                   stage_res(node,ir)=ells(node)+depsw
                   IF (trig_res2asr(ia).ge.ells(node)) THEN
                      avail_vol_to_asr(ia,ij)=avail_vol_to_asr(ia,ij)+
     &                (stage_res(node,ir)-trig_res2asr(ia))*sfactor(ia)
     &                *gdar/43560.
                   ELSE
                     avail_vol_to_asr(ia,ij)=avail_vol_to_asr(ia,ij)+
     &                (stage_res(node,ir)-ells(node)+(ells(node)-
     &                trig_res2asr(ia))*s(node))*gdar*sfactor(ia)/43560.
                   END IF
                ELSE
                   IF (trig_res2asr(ia).lt.ells(node)) THEN
```

```
                     avail_vol_to_asr(ia,ij)=avail_vol_to_asr(ia,ij)+
     &               (stage1-trig_res2asr(ia))*s(node)*afact*gdar
     &               /(43560.*noresincell(node))
                  END IF
               ENDIF
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
               IF (pond_in_res(node,ir) .gt. 0.000001) THEN
                  afact = sfactor(ia)
                  depsw = pond_in_res(node,ir)
                  stage_res(node,ir) = ells_in_res(node,ir) + depsw
                  IF (trig_res2asr(ia).ge.ells_in_res(node,ir))
     &            THEN
                     avail_vol_to_asr(ia,ij)=avail_vol_to_asr(ia
     &               ,ij)+(stage_res(node,ir)-trig_res2asr(ia))*
     &               afact*gdar/43560.
                  ELSE
                     avail_vol_to_asr(ia,ij)=avail_vol_to_asr(ia
     &               ,ij)+(stage_res(node,ir)-ells_in_res(node,ir)+
     &               (ells_in_res(node,ir)-trig_res2asr(ia))
     &               *s(node))*gdar*afact/43560.
                  END IF
               END IF
            ENDIF
            ENDIF
          END DO
          END DO
        END DO
      END IF
!
      DO ij=1,nasr_from_res(ia)
        avail_vol_to_asr(ia,ij)=amax1(avail_vol_to_asr(ia,ij),0.0)
      END DO
!
!   inject excess water from reservoir to asr
!
      IF (nasr_from_res(ia).gt.0) THEN
        DO j=1,nasr_from_res(ia)
          vol_to_asr_frm_res(j)=0.0
          vol_in=avail_vol_to_asr(ia,j)
          CALL asr (asrnum(2,ia,j,1), 1, vol_in, vol_not_inj, j,
     &       tot_cnl_dmd_asr)
          vol_to_asr_frm_res(j)=vol_to_asr_frm_res(j)+vol_in-
     &       vol_not_inj
          jstr1=jstr1+1
          strflw(jstr1,idss)=vol_to_asr_frm_res(j)/1.9835
        END DO
      END IF
!
!   adjust water levels in reservoir
!
      IF (nasr_from_res(ia).gt.0) THEN
        DO j=1,nasr_from_res(ia)
          DO k=1,nnodes_res_to_asr(ia,j)
            node=icol_asr(ia,j,k)-minx(irow_asr(ia,j,k))+1+
     &         isum(irow_asr(ia,j,k))
```

```
              DO ir = 1,noresincell(node)
              IF (ia .eq. ires_index_in_cell(node,ir)) THEN
              IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &        (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
                dpth_red_due_to_inj=vol_to_asr_frm_res(j)*43560.
     &          /(gdar*nnodes_res_to_asr(ia,j)*(1.0-carfact(node)))
                rem_dpth_red=amax1(dpth_red_due_to_inj-pond(node)
     &          ,0.0)
                pond(node)=pond(node)-amin1(dpth_red_due_to_inj,
     &          pond(node))
                rchg(node)=rchg(node)-rem_dpth_red
              ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
                dpth_red_due_to_inj=vol_to_asr_frm_res(j)*43560.
     &          /(nnodes_res_to_asr(ia,j)*sfactor(ia)*gdar)
                rem_dpth_red=amax1(dpth_red_due_to_inj
     &          -pond_in_res(node,ir),0.0)
                stage_res(node,ir) = stage_res(node,ir)
     &          - rem_dpth_red/s(node) - amin1(pond_in_res(node,ir)
     &          ,dpth_red_due_to_inj)
                pond_in_res(node,ir) = pond_in_res(node,ir)
     &          - amin1(dpth_red_due_to_inj,pond_in_res(node,ir))
                res_to_asr(ia) = res_to_asr(ia) + dpth_red_due_to_inj
     &          *sfactor(ia)*gdar/43560.
              ENDIF
              ENDIF
              END DO
            END DO
          END DO
        END IF
!
!
!   recover water from asr to reservoir
!
        IF (nasr_to_res(ia).gt.0) THEN
          DO j=1,nasr_to_res(ia)
            vol_to_res_frm_asr(j)=0.0
            IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &      (ia) .eq. 'NO ') THEN
               stage_recov=h(nodewa(ia))+pond(nodewa(ia))
               if ( stage_recov .lt. trig_resfasr(ia,1)
     $          .and. stage_recov .gt. trig_resfasr(ia,2)) then
                vol_in = asrvol(asrnum(2,ia,j,2))
               else
                vol_in = 0.0
               endif
            ELSE
               DO ir = 1,noresincell(node)
                IF (ia .eq. ires_index_in_cell(node,ir)) THEN
                  stage_recov = stage_res(node,ir)
                  if ( stage_recov .lt. trig_resfasr(ia,1)
     $             .and. stage_recov .gt. trig_resfasr(ia,2)) then
                   vol_in = asrvol(asrnum(2,ia,j,2))
                  else
```

```
                  vol_in = 0.0
                endif
              ELSE
                vol_in=0.0
              ENDIF
            ENDDO
          ENDIF
          CALL asr (asrnum(2,ia,j,2), 2, vol_in, bal_vol, j,
     &       tot_cnl_dmd_asr)
          vol_to_res_frm_asr(j)=vol_to_res_frm_asr(j)+vol_in-bal_vol
          jstr1=jstr1+1
          strflw(jstr1,idss)=vol_to_res_frm_asr(j)/1.9835
        END DO
      END IF
!
!   adjust water levels in reservoir
!
      IF (nasr_to_res(ia).gt.0) THEN
        DO j=1,nasr_to_res(ia)
          DO k=1,nnodes_res_from_asr(ia,j)
            node=icol_asr_rec(ia,j,k)-minx(irow_asr_rec(ia,j,k))+1+
     &         isum(irow_asr_rec(ia,j,k))
            DO ir = 1,noresincell(node)
            IF (ia .eq. ires_index_in_cell(node,ir)) THEN
            IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &        (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
              dpth_inc_due_to_rec=vol_to_res_frm_asr(j)*43560.
     &         /(gdar*nnodes_res_from_asr(ia,j)*(1.0-carfact(node)))
              equiv_depth_to_wt=amax1(ells(node)-h(node),0.0)*s(node)
              rchg(node)=rchg(node)+amin1(equiv_depth_to_wt,
     &         dpth_inc_due_to_rec)
              pond(node)=pond(node)+amax1(dpth_inc_due_to_rec-
     &         equiv_depth_to_wt,0.0)
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
              dpth_inc_due_to_rec=vol_to_res_frm_asr(j)*43560.
     &        /(gdar*sfactor(ia)*nnodes_res_from_asr(ia,j))
              equiv_depth_to_wt=amax1(ells_in_res(node,ir)
     &        - stage_res(node,ir),0.0)*s(node)
              stage_res(node,ir) = stage_res(node,ir)
     &        + amin1(equiv_depth_to_wt,dpth_inc_due_to_rec)
     &        / s(node) + amax1(dpth_inc_due_to_rec
     &        -equiv_depth_to_wt,0.0)
              pond_in_res(node,ir) = pond_in_res(node,ir)
     &        + amax1(dpth_inc_due_to_rec-equiv_depth_to_wt,0.0)
              res_from_asr(ia) = res_from_asr(ia)
     &        + dpth_inc_due_to_rec*sfactor(ia)*gdar/43560.
            ENDIF
            ENDIF
            END DO
          END DO
        END DO
      END IF
      DO 40 i=1,noutstr(ia)
```

```
          IF (res_out_type(ia,i).eq.'WSPLY '.or.res_out_type(ia,i).eq.'S
     &EEPG') GO TO 40
!
          IF (inodopt(ia,i,2).eq.1) THEN
            IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &      (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
               head_g=h(idnstrm(ia,i))+pond(idnstrm(ia,i))
               afact=1.0
               IF (head_g.ge.ells(idnstrm(ia,i))) THEN
                  IF (head_g.gt.ells(idnstrm(ia,i))) THEN
                    afact=total_afact(idnstrm(ia,i))
                  END IF
                  IF (afact.le.1.0) THEN
                    sgn=1
                  ELSE
                    sgn=-1
                  END IF
                  adjust_for_ovlf=sum_chg_dpth_ovl_flw(idnstrm(ia,i))
                  term1_for_adjust_area=(sumet_outside_res(idnstrm(ia,i)
     &            )+seep_loss_res(idnstrm(ia,i))-sumrain(idnstrm(ia,i))
     &            /12.0-sum_chg_depth_levee_spg(idnstrm(ia,i)))
     &            *((1.0-afact)/ afact)
                  term2_for_adjust_area=(outf_fc_frm_res_to_cell(idnstrm
     &            (ia,i))+daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))+
     &            outf_seep_loc_to_res_cell(idnstrm(ia,i))
     &            +adjust_for_ovlf)/afact
                  head=ells(idnstrm(ia,i))+pond(idnstrm(ia,i))/afact+
     &            term1_for_adjust_area-adjust_dur_excess(idnstrm(ia,i))
     &            -(outf_fc_frm_res_to_cell(idnstrm(ia,i))+
     &            daily_outf_fc_frm_res_to_cell1(idnstrm(ia,i))+
     &            outf_seep_loc_to_res_cell(idnstrm(ia,i)))/afact-
     &            sum_chg_dpth_ovl_flw(idnstrm(ia,i))/afact
                  IF (idnstrm(ia,i).eq.nodepw(ia,i,1).and.inodopt(ia,i,
     &            3).eq.1) THEN
                     head_avvol=head+
     &               daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))/afact
                  ELSE
                     head_avvol=head-adjust_for_area_m1(idnstrm(ia,i))
                  END IF
!
                 head=head-adjust_for_area_m1(idnstrm(ia,i))
                 IF (idnstrm(ia,i).eq.nodepw(ia,i,1).and.inodopt(ia,i,3)
     &            .eq.1) THEN
                    head_avvol=head+
     &              daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))/afact
                 ELSE
                    head_avvol=head
                 END IF
               ELSE
                  head=head_g
                  head_avvol=head_g
               END IF
               head=amax1(head,ells(idnstrm(ia,i)))
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
```

```
c
              DO ir = 1,noresincell(idnstrm(ia,i))
                IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir)) THEN
                   head=stage_res(idnstrm(ia,i),ir)
                   head_avvol=head
                ENDIF
              ENDDO
            ENDIF
c       if (resname(ia).eq.'STA1E ') print *,POND(IDNSTRM(IA,I))
c    $,SUMRAIN(IDNSTRM(IA,I))/12.0,SUMET(IDNSTRM(IA,I))
c    $,seep_loss_res(idnstrm(ia,i)),sum_chg_depth_levee_spg(IDNSTRM
c    $(IA,I)),outf_fc_frm_res_to_cell(IDNSTRM(IA,I))
c    $+daily_outf_fc_frm_res_to_cell1(IDNSTRM(IA,I))
c    $,adjust_for_area_m(IDNSTRM(IA,I)),adjust_dur_excess(idnstrm(ia,i))
c    $,HEAD,'   STAOUT  ',sum_chg_dpth_ovl_flw(IDNSTRM(IA,I))
c    $,HEAD_G,daily_excess_adjust(IDNSTRM(IA,I)),adjust_for_ovlf
c    $,term1_for_adjust_area,term2_for_adjust_area,HEAD_avvol
          ELSE
             head=chdep(ioutcnl(ia,i))
          END IF
!
          IF (inodopt(ia,i,3).eq.1) THEN
            IF (((sfactor(iwmads(ia,i)) .GE. sfactmin .or.ires_small_sim
     &      (iwmads(ia,i)) .eq. 'NO ') .AND. iwmads(ia,i) .le. ntotres)
     &      .OR. iwmads(ia,i) .gt. ntotres) THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
            deptgw =(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))
     &      * s(nodepw(ia,i,1))
            IF (pond(nodepw(ia,i,1)).gt. deptgw) THEN
              tail = pond(nodepw(ia,i,1)) - deptgw
     &        + ells(nodepw(ia,i,1))
            ELSE
              tail = (pond(nodepw(ia,i,1))-deptgw)
     &        /s(nodepw(ia,i,1)) + ells(nodepw(ia,i,1))
            ENDIF
            IF (iwmads(ia,i).gt.ntotres.or.tail.le.ells(nodepw(ia,i,1)))
     &        THEN
              afact=1.0
            ELSE
              afact=total_afact(nodepw(ia,i,1))
            END IF
            IF (afact.le.1.0) THEN
              sgn=1
            ELSE
              sgn=-1
            END IF
            adjust_for_ovlf=sum_chg_dpth_ovl_flw(nodepw(ia,i,1))
            term1_for_adjust_area=(sumet_outside_res(nodepw(ia,i,1))+
     &       seep_loss_res(nodepw(ia,i,1))-sumrain(nodepw(ia,i,1))/12.0-
     &       sum_chg_depth_levee_spg(nodepw(ia,i,1)))
     &       *((1.0-afact)/afact)
            term2_for_adjust_area=(outf_fc_frm_res_to_cell(nodepw(ia,i,
     &       1))+daily_outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &       outf_seep_loc_to_res_cell(nodepw(ia,i,1))+adjust_for_ovlf)/
     &       afact
            IF (tail.gt.ells(nodepw(ia,i,1)).and.iwmads(ia,i)
```

```
     &        .le.ntotres.and.idnstrm(ia,i).ne.nodepw(ia,i,1)) THEN
              IF (sgn*
     &         term1_for_adjust_area.lt.term2_for_adjust_area.and.tail
     &         .gt.ells(nodepw(ia,i,1))) THEN
               adjust_dur_excess(nodepw(ia,i,1))=
     &          adjust_for_area_m(nodepw(ia,i,1))+
     &          adjust_dur_deficit(nodepw(ia,i,1))
              ELSE
               adjust_dur_deficit(nodepw(ia,i,1))=
     &          adjust_dur_excess(nodepw(ia,i,1))
               adjust_dur_excess(nodepw(ia,i,1))=
     &          adjust_dur_deficit(nodepw(ia,i,1))
              END IF
              tail=ells(nodepw(ia,i,1))+pond(nodepw(ia,i,1))/afact+
     &         term1_for_adjust_area-adjust_dur_excess(nodepw(ia,i,1))-
     &         (outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &         outf_seep_loc_to_res_cell(nodepw(ia,i,1)))/afact-
     &         sum_chg_dpth_ovl_flw(nodepw(ia,i,1))/afact+
     &         daily_chg_dpth_ovl_vol1(nodepw(ia,i,1))/afact
              IF (sgn*
     &         term1_for_adjust_area.ge.term2_for_adjust_area.or.tail.le
     &         .ells(nodepw(ia,i,1))) THEN
!
               adjust_for_area_m(nodepw(ia,i,1))=term1_for_adjust_area-
     &          term2_for_adjust_area+daily_excess_adjust(nodepw(ia,i,
     &          1))
               IF (afact.le.1.0) THEN
                 adjust_for_area_m(nodepw(ia,i,1))=
     &            amax1(adjust_for_area_m(nodepw(ia,i,1)),0.0)
               END IF
               adjust_for_area_m(nodepw(ia,i,1))=sgn*amin1(sgn*
     &          adjust_for_area_m(nodepw(ia,i,1)),sgn*(tail-
     &          (ells(nodepw(ia,i,1))+pond(nodepw(ia,i,1)))))
!
               IF (h(nodepw(ia,i,1))+pond(nodepw(ia,i,1))
     &          .le.ells(nodepw(ia,i,1)).and.iwmads(ia,i).le.ntotres)
     &          THEN
                 IF (tail.le.ells(nodepw(ia,i,1))+pond(nodepw(ia,i,1)))
     &             THEN
                   adjust_for_area_m(nodepw(ia,i,1))=
     &              daily_et(nodepw(ia,i,1))+
     &              daily_seep_loss_res(nodepw(ia,i,1))+
     &              chg_depth_levee_spg(nodepw(ia,i,1))-
     &              daily_outf_fc_frm_res_to_cell(nodepw(ia,i,1))-
     &              daily_outf_seep_loc_to_res_cell(nodepw(ia,i,1))-
     &              rain(nodepw(ia,i,1))/12.0
                   adjust_for_area_m(nodepw(ia,i,1))=
     &              amax1(adjust_for_area_m(nodepw(ia,i,1)),0.0)
                   adjust_for_area_m(nodepw(ia,i,1))=
     &              amin1(adjust_for_area_m(nodepw(ia,i,1)),tail-
     &              (ells(nodepw(ia,i,1))+pond(nodepw(ia,i,1))))
                 END IF
               END IF
!
               tail=tail-adjust_for_area_m(nodepw(ia,i,1))
              END IF
            END IF
```

```
            IF (tail.gt.ells(nodepw(ia,i,1)).or.pond(nodepw(ia,i,1))+
     &       h(nodepw(ia,i,1)).gt.ells(nodepw(ia,i,1))) THEN
              IF (nodepw(ia,i,1).eq.idnstrm(ia,i) .and.
     &        (sfactor(ia) .GE. sfactmin .or.ires_small_sim(ia) .eq.
     &        'NO ')) THEN
                tail=est_cell_stg_o_res(nodepw(ia,i,1))
              END IF
            END IF
c        if (resname(ia).eq.'S332BR') print *,TAIL,'  TAIL FOR OUTF'
c           IF (nodepw(ia,i,1).ne.idnstrm(ia,i)) tail=amax1(tail,
c    &       ells(nodepw(ia,i,1)))
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
               IF (nodepw(ia,i,1) .eq. idnstrm(ia,i)) then
                  deptgw =(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))
     &            * s(nodepw(ia,i,1))
                  IF (pond(nodepw(ia,i,1)).gt. deptgw)
     &            THEN
                     tail = pond(nodepw(ia,i,1))
     &               - deptgw + ells(nodepw(ia,i,1))
                  ELSE
                     tail = (pond(nodepw(ia,i,1))
     &               -deptgw)/s(nodepw(ia,i,1))+ells(nodepw(ia,i,1))
                  ENDIF
                  DO ir = 1,noresincell(nodepw(ia,i,1))
                     IF (iwmads(ia,i) .EQ. ires_index_in_cell
     &               (nodepw(ia,i,1),ir)) THEN
                       tail = stage_res(nodepw(ia,i,1),ir)
                     ENDIF
                  ENDDO
               ELSE
                  IF (iwmads(ia,i).gt.ntotres) then
                     deptgw =(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))
     &               * s(nodepw(ia,i,1))
                     IF (pond(nodepw(ia,i,1)).gt. deptgw) THEN
                        tail = pond(nodepw(ia,i,1)) - deptgw
     &                  + ells(nodepw(ia,i,1))
                     ELSE
                        tail = (pond(nodepw(ia,i,1))-deptgw)
     &                  /s(nodepw(ia,i,1)) + ells(nodepw(ia,i,1))
                     ENDIF
                  ELSE
                    DO ir = 1,noresincell(nodepw(ia,i,1))
                       IF (iwmads(ia,i) .EQ. ires_index_in_cell
     &                 (nodepw(ia,i,1),ir)) THEN
                         tail = stage_res(nodepw(ia,i,1),ir)
                       ENDIF
                    ENDDO
                  ENDIF
               ENDIF
            ENDIF
          ELSE
            IF (ircnl(ia,i,1).le.nch) THEN
              IF (nodcr(ircnl(ia,i,1)) .gt. 1) then
               diff_canlr_stg_frm_ds_end = float(nodcr(ircnl(ia,i,1))
     &         -NODEPOS_CANLR(ia,i,1))/float(nodcr(ircnl(ia,i,1))-1)
```

```
     &         *hdc(ircnl(ia,i,1),iseas)
               ELSE
                diff_canlr_stg_frm_ds_end = 0.0
               ENDIF
               tail=chdep(ircnl(ia,i,1))+diff_canlr_stg_frm_ds_end
             ELSE
               tail=crstelev(ia,i,2)
             END IF
           END IF
!
           IF (type(ia,i).eq.'GRAVITY') THEN
             IF (wrexp(ia,i,2).eq.1.0) THEN
                wcrsth = tail
             ELSE
                wcrsth=amax1(crstelev(ia,i,2),tail)
             ENDIF
           ELSE
             wcrsth=crstelev(ia,i,2)
           END IF
!
           IF (inodopt(ia,i,2).eq.1) THEN
              head_av = head
              DO ir = 1,noresincell(idnstrm(ia,i))
                 IF (ia .EQ. ires_index_in_cell
     &           (idnstrm(ia,i),ir)) THEN
                    head_av = (prev_stage_res(idnstrm(ia,i),ir)
     &              + head)/2.0
                 ENDIF
              ENDDO
           ELSE
              head_av = head
           ENDIF
           IF (head_av .gt.wcrsth) THEN
            IF (wrexp(ia,i,2).eq. 1.5 .and.type(ia,i).eq.'GRAVITY')
     &      THEN
             IF (wcrsth.gt.crstelev(ia,i,2)) THEN
               crelev = crstelev(ia,i,2)
               submerg_fact = (wcrsth - crelev)/(head_av - crelev)
               rel_crst_length = (head_av - crelev)/(head_av - crelev
     &         + 1.0)
               rmod_limit = 0.85 - 0.50*rel_crst_length
               reduc_fact = (submerg_fact -rmod_limit)
     &         /(1.0 -rmod_limit)
               reduc_fact = amax1(reduc_fact,0.0)
               reduc_fact = (1.0 - reduc_fact) ** 0.143
               qfactor=reduc_fact
             ELSE
               qfactor=1.0
             END IF
            ELSE
              qfactor=1.0
            ENDIF
             IF (wrexp(ia,i,2).eq.1.0) THEN
               qout=weircff(ia,i,2)*(head_av-tail)
             ELSE
               IF (wrexp(ia,i,2).eq. 1.5 .and.type(ia,i).eq.'GRAVITY')
     &         THEN
```

```
                qout=weircff(ia,i,2)*(head_av-crstelev(ia,i,2))**wrexp
     &          (ia,i,2)*qfactor
              ELSE
                qout = weircff(ia,i,2)*(head_av-wcrsth)**wrexp(ia,i,2)
              ENDIF
            END IF
            IF (wcrsth.lt.stgmx(ia)) THEN
              IF (inodopt(ia,i,2).eq.1) THEN
                IF (wrexp(ia,i,2).ne.1.0) THEN
                  IF(wrexp(ia,i,2).eq. 1.5 .and.type(ia,i).eq.'GRAVITY')
     &            THEN
                    qoutm=weircff(ia,i,2)*(stgmx(ia)-crstelev(ia,i,2))**
     &               wrexp(ia,i,2)*qfactor
                  ELSE
                    qoutm=weircff(ia,i,2)*(stgmx(ia)-wcrsth)**wrexp(ia,i
     &              ,2)
                  ENDIF
                  qoutm = amin1(qoutm,RMXCPO(ia,i))
                ELSE
                  qoutm=weircff(ia,i,2)*(stgmx(ia)-tail)
                END IF
              ELSE
                qoutm=999999.
              END IF
            ELSE
              qoutm=999999.
            END IF
            if (res_out_strname(ia,i).eq.res_struc_name_spec_code(38))
     &       then
              qoutfact = frac_s319_sta1ee
            elseif (res_out_strname(ia,i).eq.res_struc_name_spec_code
     &      (39)) then
              qoutfact = frac_s319_sta1ew
            elseif (res_out_strname(ia,i).eq.res_struc_name_spec_code
     &      (41)) then
              qoutfact = ratio_flow_s6_sta2e
            elseif (res_out_strname(ia,i).eq.res_struc_name_spec_code
     &      (42)) then
              qoutfact = ratio_flow_s6_sta2m
            elseif (res_out_strname(ia,i).eq.res_struc_name_spec_code
     &      (43)) then
              qoutfact = ratio_flow_s6_sta2w
            else
              qoutfact = 1.0
            endif
c          if (res_out_strname(ia,i).eq.res_struc_name_spec_code(41))
c    &      print *,qout,qoutm,qoutfact,RMXCPO(ia,i),wcrsth,'  LLLL  ',
c    &      stgmx(ia),head_av,res_out_strname(ia,i)
            qout=amin1(qout,qoutm)*qoutfact
          ELSE
            IF (idnstrm(ia,i).eq.nodepw(ia,i,1) .AND.
     &      wrexp(ia,i,2).eq.1.0) THEN
              qout=weircff(ia,i,2)*(head_av-tail)
            ELSE
              qout=0.0
            ENDIF
          END IF
```

```
          IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(8))
     &     THEN
            depth_below_target_3a28=stage_import_wca(iwca3a_3a28)-
     &       offset(iwca3a_3a28,1)-avg_sim_wca_stg(iwca3a_3a28)
            depth_below_target_3a28=amax1(depth_below_target_3a28,0.0)
            IF (depth_below_target_3a28.lt.0.1) THEN
              qout=10.0*depth_below_target_3a28*qout
            END IF
          END IF
!
          IF (inodopt(ia,i,2).eq.1) THEN
            gwdpth=(ells(idnstrm(ia,i))-h(idnstrm(ia,i)))*s(idnstrm(ia,
     &       i))
            IF (pond(idnstrm(ia,i)).ge.gwdpth) THEN
              stage=pond(idnstrm(ia,i))-gwdpth+ells(idnstrm(ia,i))
            ELSE
              stage=(pond(idnstrm(ia,i))-gwdpth)/s(idnstrm(ia,i))+
     &         ells(idnstrm(ia,i))
            END IF
!
            IF (idnstrm(ia,i).eq.nodepw(ia,i,1)) THEN
              if (wrexp(ia,i,2).eq.1.0) THEN
                 DO ir = 1,noresincell(idnstrm(ia,i))
                   IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir)
     &              .and. iwmads(ia,i) .gt. ntotres) THEN
c  compute equilibruim stage for grid cell-reservoir system
c
                     call reserv_cell_equil_stg(stage_res,ells_in_res
     &               ,ells,sfactor,total_small_afact,idnstrm,h,pond,s
     &               ,equil_stg,ia,ir,i,carfact)
c                    if (resname(ia).eq.'R332C ') print *,stage_res
c    &               (idnstrm(ia,i),ir),h(idnstrm(ia,i))+pond
c    &               (idnstrm(ia,i)),equil_stg,'  LSEEPVL'
                     wcrsth1 = equil_stg
                   ELSE
                     wcrsth1 = tail
                   ENDIF
                 ENDDO
              else
                wcrsth1= wcrsth
                DO ir = 1,noresincell(idnstrm(ia,i))
                  IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir)
     &            .and. iwmads(ia,i) .le. ntotres .and.
     &            (sfactor(iwmads(ia,i)) .LT. sfactmin .and.
     &            ires_small_sim(iwmads(ia,i)) .eq. 'YES') .and.
     &            wrexp(ia,i,2).ne.1.0) then
                    do irdn = 1,noresincell(nodepw(ia,i,1))
                       if (iwmads(ia,i) .eq.ires_index_in_cell
     &                 (nodepw(ia,i,1),irdn)) then
                        dpth_to_crest =(min(crstelev(ia,i,2),ells_in_res
     &                  (nodepw(ia,i,1),irdn))-min(tail,ells_in_res
     &                  (nodepw(ia,i,1),irdn)))* s(nodepw(ia,i,1)) +
     &                  max(crstelev(ia,i,2),ells_in_res(nodepw(ia,i,1)
     &                  ,irdn))-max(ells_in_res(nodepw(ia,i,1),irdn)
     &                  ,tail)
                        head_vollim = head - max(dpth_to_crest,0.0)
     &                  * sfactor(iwmads(ia,i))/sfactor(ia)
```

```
                        if (head_vollim .gt. crstelev(ia,i,2)) then
                           head_equil = (head_vollim*sfactor(ia)+
     &                     max(tail,ells_in_res(nodepw(ia,i,1),irdn))
     &                     *sfactor(iwmads(ia,i)))/(sfactor(ia)
     &                     +sfactor(iwmads(ia,i)))
                           wcrsth1 = head_equil
                        endif
                       endif
                    enddo
                   ENDIF
                 ENDDO
               endif
             ELSE
               wcrsth1=wcrsth
               DO ir = 1,noresincell(idnstrm(ia,i))
                 IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir)
     &           .and. iwmads(ia,i) .le. ntotres .and.
     &           (sfactor(iwmads(ia,i)) .LT. sfactmin .and.
     &           ires_small_sim(iwmads(ia,i)) .eq. 'YES')
     &           .and. wrexp(ia,i,2).ne.1.0) then
                   do irdn = 1,noresincell(nodepw(ia,i,1))
                      if (iwmads(ia,i) .eq.ires_index_in_cell
     &                (nodepw(ia,i,1),irdn)) then
                         dpth_to_crest =(min(crstelev(ia,i,2),ells_in_res
     &                   (nodepw(ia,i,1),irdn))-min(tail,ells_in_res
     &                   (nodepw(ia,i,1),irdn)))* s(nodepw(ia,i,1)) +
     &                   max(crstelev(ia,i,2),ells_in_res(nodepw(ia,i,1)
     &                   ,irdn))-max(ells_in_res(nodepw(ia,i,1),irdn)
     &                   ,tail)
                         head_vollim = head - max(dpth_to_crest,0.0)
     &                    * sfactor(iwmads(ia,i))/sfactor(ia)
                         if (head_vollim .gt. crstelev(ia,i,2)) then
                            head_equil = (head_vollim*sfactor(ia)+
     &                     max(tail,ells_in_res(nodepw(ia,i,1),irdn))
     &                     *sfactor(iwmads(ia,i)))/(sfactor(ia)
     &                     +sfactor(iwmads(ia,i)))
                            wcrsth1 = head_equil
                         endif
c                        if (resname(ia).eq.'R332C ') print *,
c    &                    resname(iwmads(ia,i)),head,head_vollim
c    &                     ,crstelev(ia,i,2),head_equil,tail
c    &                     ,ells_in_res(nodepw(ia,i,1),irdn)
c    &                     ,dpth_to_crest,'  R332COT  ',wcrsth1
c    &                     ,head_av
                      endif
                   enddo
                 ENDIF
               ENDDO
             END IF
             IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &       (ia) .eq. 'NO ') THEN
c  reservoir is large enough not to treat as separate entity in
c  grid domain
               IF (stage.gt.ells(idnstrm(ia,i))) THEN
                 IF (idnstrm(ia,i).eq.nodepw(ia,i,1)) THEN
                   avvolgr(i)=(head_avvol-wcrsth1)*gdar*sfactor(ia)
     &             /86400.*0.90
```

```
                ELSE
                  avvolgr(i)=(head-wcrsth1)*gdar*sfactor(ia)/86400.
     &            *0.95
                END IF
              ELSE
                avvolgr(i)=(ells(idnstrm(ia,i))-wcrsth1)*gdar/86400.*
     &          sfactor(ia)-(ells(idnstrm(ia,i))-stage)*
     &          s(idnstrm(ia,i))*gdar/86400.*0.90
              END IF
              signh = 1.0
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
              DO ir = 1,noresincell(idnstrm(ia,i))
                IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir)) THEN
                  IF (idnstrm(ia,i).eq.nodepw(ia,i,1) .AND.
     &              wrexp(ia,i,2).eq.1.0) THEN
                    IF (head .GE. wcrsth1) THEN
                       signh = 1.0
                    ELSE
                       signh = -1.0
                    ENDIF
                    rlim_frac = frac_seep_max_lim(ia)
                  ELSE
                    signh = 1.0
                    rlim_frac = 0.95
                  ENDIF

                  IF (head .gt. ells_in_res(idnstrm(ia,i),ir)) THEN
                    IF (wcrsth1 .ge.ells_in_res(idnstrm(ia,i),ir)) THEN
                      avvolgr(i)=(head-wcrsth1)*gdar*sfactor(ia)/86400.
                    ELSE
                      avvolgr(i)=(head-ells_in_res(idnstrm(ia,i),ir))
     &                *gdar*sfactor(ia)/86400. + (ells_in_res
     &                (idnstrm(ia,i),ir)-wcrsth1)*s(idnstrm(ia,i))
     &                *gdar*sfactor(ia)/86400.
                    ENDIF
                  ELSE
                    IF (wcrsth1 .ge.ells_in_res(idnstrm(ia,i),ir)) THEN
                      avvolgr(i)=(ells_in_res(idnstrm(ia,i),ir)-wcrsth1
     &                )*gdar/86400.*sfactor(ia)-(ells_in_res(idnstrm(ia
     &                ,i),ir)-head)*s(idnstrm(ia,i))*gdar*sfactor(ia)
     &                /86400.
                    ELSE
                      avvolgr(i)= (head - wcrsth1) * s(idnstrm(ia,i))
     &                *gdar*sfactor(ia)/86400.
                    ENDIF
                  ENDIF
                  avvolgr(i) = avvolgr(i) * rlim_frac * signh
                ENDIF
              ENDDO
            ENDIF
            avvolgr(i)=amax1(avvolgr(i),0.0)
            avvolgr_mx(ia,i) = avvolgr(i)
          ELSE
            crest_elev_struc = crstelev(ia,i,2)
            avvolgr(i)=(chdep(ioutcnl(ia,i))-crest_elev_struc)*
```

```
     &       cl(ioutcnl(ia,i))*width(ioutcnl(ia,i))+cgseep(ioutcnl(ia,i)
     &       ,crest_elev_struc,0,0.)+tovflc(ioutcnl(ia,i),
     &       crest_elev_struc,0,0.,0,1)+qu(ioutcnl(ia,i))
             avvolgr(i)=amax1(avvolgr(i),0.0)/86400.
             signh = 1.0
           END IF
!
           IF (iwmads(ia,i).le.ntotres) THEN
             resoutmx=amax1(volwtr(iwmads(ia,i)),0.0)
           ELSE
             resoutmx=9999999.
           END IF
!
           DO ip=1,npumptores
             IF (iwmads(ia,i).le.ntotres) THEN
               IF (ibsn_no_res(iwmads(ia,i)).eq.ibsnres(ip)) THEN
                 resoutmx=amin1(pumpcap_res(ip),amax1(volwtr(iwmads(ia,i)
     &            ),0.0))
               END IF
             ELSE
               IF (iwmads(ia,i).eq.ibsnres(ip)) THEN
                 resoutmx=pumpcap_res(ip)
               END IF
             END IF
           END DO
!
           qoutflw(ia,i)=signh*amin1(signh*qout,avvolgr(i),resoutmx)
c          if (res_out_strname(ia,i).eq.'ST3OT1') print *,qout,avvolgr(i)
c    &     ,qoutm,capc,head,head_av,tail,qoutflw(ia,i),'  ST3OT1'
c          if (resname(ia).eq.'STA1W ') print *,qout,avvolgr(i),qoutm
c    &      ,capc,'  STA1W ',wcrsth1,head,head_av,pond(idnstrm(ia,i))
c    &      ,qoutflw(ia,i),resoutmx,res_out_strname(ia,i)
c    &      ,month,iday
           IF (inodopt(ia,i,3).ne.1.and.ircnl(ia,i,1).le.nch) THEN
             IF (itrgcnl(ia,i,1) .le. nch) THEN
              if (nodcr(itrgcnl(ia,i,1)).gt.1) then
                diff_stg_cnl_trg = float((nodcr(itrgcnl(ia,i,1))-
     &          nodepos_rcnl_trg(ia,i,1)))/float(nodcr(itrgcnl
     &          (ia,i,1))-1)*hdc(itrgcnl(ia,i,1),iseas)
              else
                diff_stg_cnl_trg = 0.0
              endif
              dn_cnl_stg_trg = chdep(itrgcnl(ia,i,1))+diff_stg_cnl_trg
             ELSE
              if (icell_trg_loc(ia,i,1) .gt. 0) THEN
                 dn_cnl_stg_trg = h(icell_trg_loc(ia,i,1))
     &           + pond(icell_trg_loc(ia,i,1))
              else
                 dn_cnl_stg_trg = -901.0
              endif
             ENDIF
             IF (dn_cnl_stg_trg.lt.dwnstgmx(ia,i,1)+
     &       offset_reg_res(ia,i,1).and.dn_cnl_stg_trg
     &       .ge.dwnstgmx(ia,i,1)) THEN
              qoutflw(ia,i)=1.0/offset_reg_res(ia,i,1)*(dwnstgmx(ia,i,1)
     &         +offset_reg_res(ia,i,1)-dn_cnl_stg_trg)*qoutflw(ia,
     &         i)
```

```
            ELSE IF (dn_cnl_stg_trg.ge.dwnstgmx(ia,i,1)+
     &       offset_reg_res(ia,i,1)) THEN
              qoutflw(ia,i)=0.0
            END IF
c           if (resname(ia).eq.'C4IMP') print *,qoutflw(ia,i)
c    $        ,dn_cnl_stg_trg,dwnstgmx(ia,i,1),offset_reg_res(ia,i,1)
c    $        ,cnm(itrgcnl(ia,i,1)),hdc(itrgcnl(ia,i,1),iseas)
c    $        ,chdep(itrgcnl(ia,i,1)),nodepos_rcnl_trg(ia,i,1)
c    $        ,diff_stg_cnl_trg,nodcr(itrgcnl(ia,i,1))
          ELSEIF (inodopt(ia,i,3).eq.1) then
            nrestrt = 0
            do kk = 1,nrgtrg(ia,i)
              if (h(noder_trg(ia,i,kk))+pond(noder_trg(ia,i,kk))
     &        .gt. stgr_trg(ia,i,kk)) then
                 nrestrt = nrestrt + 1
              endif
            enddo
            if (nrestrt .gt. 0) qoutflw(ia,i)=0.0
          END IF
          IF (resname(ia).eq.name_res_spec_code(1)) THEN
            IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(4)
     &       .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(5))
     &       THEN
              IF (lwslo(iewca3b)) THEN
                dischg_to_s8_frm_sta34=amin1(qoutflw(ia,i),s8capac)+
     &           dischg_to_s8_frm_sta34
                qoutflw(ia,i)=amin1(qoutflw(ia,i),s8capac)
                s8capac=s8capac-qoutflw(ia,i)
              ELSE
                qoutflw(ia,i)=0.0
              END IF
            ELSE IF (res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(1).or.res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(2).or.res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(3)) THEN
              IF (((.not.lwslo(inewca3a).and..not.lwslo(iewca3b))
     &         .or.(.not.lwslo(inewca3a)
     &         .and.opt_outflow_from_res_to_wca.eq.'NONL'))
     &         .and.opt_outflow_from_res_to_wca.ne.'FLDC') THEN
                qoutflw(ia,i)=0.0
              END IF
            ELSE IF (res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(6).or.res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(7)) THEN
              dischg_to_s7_frm_sta34=amin1(qoutflw(ia,i),s7capac)+
     &         dischg_to_s7_frm_sta34
              qoutflw(ia,i)=amin1(qoutflw(ia,i),s7capac)
              s7capac=s7capac-qoutflw(ia,i)
            ELSE IF (res_out_strname(ia,i)
     &       .eq.res_struc_name_spec_code(8)) THEN
              s150cap=-1.0
              stg_canl_ds_s150 = chdep(ica3)
              CALL eaaspill (s150cap, s7_spill_hw, stg_canl_ds_s150, 8,
     &         ifacteaa)
              IF (lwslo(iwca3a_3a28)) THEN
                dischg_to_ca3a=amin1(qoutflw(ia,i),s150cap-kflo(ifls150)
     &           )
```

```
          ELSE
            dischg_to_ca3a=0.0
          END IF
          qoutflw(ia,i)=dischg_to_ca3a
        ELSE IF (res_out_strname(ia,i)
&        .eq.res_struc_name_spec_code(9)) THEN
          s150cap=-1.0
          stg_canl_ds_s150 = chdep(ica3)
          CALL eaaspill (s150cap, s7_spill_hw, stg_canl_ds_s150, 8,
&          ifacteaa)
          dishg_capac=s150cap-kflo(ifls150)
          dishg_capac=amax1(dishg_capac,0.0)
          qoutflw(ia,i)=amin1(dishg_capac,qoutflw(ia,i))
        END IF
      END IF
      IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(11)
&      .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(12))
&      THEN
        CALL bpts (iday, month, 10.0, 8, 1, 1, roten_sched)
        IF (rotenenv.eq.'TRUE ') THEN
          roten_stage=avg_sim_wca_stg_out(iroten)
          roten_outflw_stage=stage_outflw_wca(iroten)
        ELSE
          roten_outflw_stage=roten_sched+offset_to_sched(8)
        END IF
        IF (rotenenv.eq.'FALSE') THEN
          IF (roten_stage.gt.roten_outflw_stage+0.1) THEN
            dischg_rot_struc=qoutflw(ia,i)
          ELSE IF (roten_stage.gt.roten_outflw_stage) THEN
            dischg_rot_struc=10.0*(roten_stage-roten_outflw_stage)*
&            qoutflw(ia,i)
          ELSE
            dischg_rot_struc=0.0
          END IF
        ELSE
          IF (roten_stage.gt.roten_outflw_stage+offset_out(iroten,2)
&          ) THEN
            dischg_rot_struc=qoutflw(ia,i)
          ELSE IF (roten_stage.gt.roten_outflw_stage+
&          offset_out(iroten,1)) THEN
            dischg_rot_struc=1.0/(offset_out(iroten,2)-
&            offset_out(iroten,1))*(roten_stage-roten_outflw_stage-
&            offset_out(iroten,1))*qoutflw(ia,i)
          ELSE
            dischg_rot_struc=0.0
          END IF
        END IF
        qoutflw(ia,i)=dischg_rot_struc
        total_outflow_from_roten=total_outflow_from_roten+
&        dischg_rot_struc
      ELSE IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(10)
&      ) THEN
        CALL bpts (iday, month, 10.0, 8, 1, 1, roten_sched)
        IF (rotenenv.eq.'TRUE ') THEN
          roten_stage=avg_sim_wca_stg_out(iroten)
          roten_outflw_stage=stage_outflw_wca(iroten)
        ELSE
```

```
          roten_outflw_stage=roten_sched+offset_to_sched(8)
        END IF
        IF (roten_stage.gt.roten_outflw_stage) THEN
          avvol=(chdep(ioutcnl(ia,i))-roten_outflw_stage)*
&          cl(ioutcnl(ia,i))*width(ioutcnl(ia,i))+cgseep(ioutcnl(ia,
&          i),roten_outflw_stage,0,0.)+tovflc(ioutcnl(ia,i),
&          roten_outflw_stage,0,0.,0,1)+qu(ioutcnl(ia,i))
          avvol=amax1(avvol,0.0)/86400.
          dischg_rot_struc=amin1(qoutflw(ia,i),avvol,qu(ioutcnl(ia,
&          i))/86400.)
        ELSE
          dischg_rot_struc=amin1(canal_capac_roten,190.0,
&          sem_dmnd_orig/1.9835,outflow_to_canal_roten,
&          qu(ioutcnl(ia,i))/86400.)
        END IF
        total_outflow_from_roten=total_outflow_from_roten+
&        dischg_rot_struc
        qoutflw(ia,i)=dischg_rot_struc
      ELSE IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(13)
&      .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(14))
&      THEN
        total_outflow_from_roten=qoutflw(ia,i)+
&        total_outflow_from_roten
        outflow_from_roten_to_west=0.0
      ELSE IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(18)
&      ) THEN
        change_lok_stg=stagelo-rstgpls(iplsday)
        IF (change_lok_stg.lt.0.0.and.stagelo.lt.stgref_asr-0.25)
&        THEN
          dischg_overspill=qoutflw(ia,i)
        ELSE
          dischg_overspill=0.0
        END IF
        qoutflw(ia,i)=dischg_overspill
      ELSE IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(33)
&      .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(34))
&      THEN
        IF (.not.lwslo(i_ds_target_indx(ia,i))) THEN
          dischg_to_bay=qoutflw(ia,i)
        ELSE
          dischg_to_bay=0.0
        END IF
        qoutflw(ia,i)=dischg_to_bay
      ELSE IF (res_out_strname(ia,i).eq.'ACMBLD') THEN
        IF (resname(iwmads(ia,i)).eq.'NORES') THEN
          avail_cap_acmbld=amax1(qoutflw(ia,i)-
&          flow_to_ca1_frm_acmebsn,0.0)
          dischg_bld=amin1(avail_cap_acmbld,amax1(qout-
&          ws_from_acmres_lwdd-avail_vol_locfc-
&          flow_to_ca1_frm_acmebsn,0.0))
          dischg_bld_acmres=dischg_bld
        ELSE
          dischg_bld=amin1(qoutflw(ia,i),weircff(iwmads(ia,i),1,1)-
&          flow_to_cpbres_frm_e1,volwtr(iwmads(ia,i)))
          dischg_bld_acmres=0.0
        END IF
        qoutflw(ia,i)=dischg_bld
```

```
          ELSE IF (res_out_strname(ia,i).eq.'C11RO ') THEN
            CALL match ('NLBWRA', 1, 6, resname, ntotres+1, 6, isowra)
            CALL match ('LBLTN', 1, 5, cnm, nch+1, 5, ilbtn)
            CALL match ('C9RES ', 1, 6, resname, ntotres+1, 6, ic9res)
            CALL match ('US27S', 1, 5, cnm, nch+1, 5, ius27s)
            istrc=0
            DO is=1,nstrctr(ius27s)
              istrc=istrc+1
              IF (flnm(istr(ius27s,is)).eq.'C11WRA') GO TO 10
            END DO
   10       IF (ilbtn.ne.0.and.ic9res.ne.0) THEN
              IF (isowra.ne.0) THEN
                rlkbltn_stor=(4.88-chdep(ilbtn))*cl(ilbtn)*width(ilbtn)
                rlkbltn_stor=amax1(rlkbltn_stor,0.0)/86400.
                rlimit_flow=amin1(discoef(ius27s,istrc,1,1),
     &           amax1(volwtr(isowra),0.0))+amax1(volwtr(ic9res),0.0)+
     &           rlkbltn_stor
                dischg_frm_res=amin1(qoutflw(ia,i),rlimit_flow)
                qoutflw(ia,i)=dischg_frm_res
              END IF
            END IF
          ELSE IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(35)
     &     ) THEN
            dischg_to_bay_s29_frm_c11r=qoutflw(ia,i)
          ELSEIF (res_out_strname(ia,i) .eq. 'NLBC9R') THEN
           avail_cap = amax1(qoutflw(ia,i) - tot_flw_to_c9r_frm_nlkblt
     $          - flow_to_c9r_frm_c9 ,0.0)
           if (nint(flow_to_lbn_frm_c11) .gt. 0) then
            cout = amin1(avail_cap,volwtr(iwmads(ia,i))
     $                  ,flow_to_lbn_frm_c11)
           else
            cout = 0.0
           endif
           qoutflw(ia,i) = cout
          ELSE IF (res_out_strname(ia,i).eq.'EART2A') THEN
            ntarg=0
            DO itarg=1,no_targ_loc(is8bsn-6)
              IF (lwslo_eaares(itarg_area(itarg,is8bsn-6))) THEN
                ntarg=ntarg+1
              END IF
            END DO
!
            IF (ntarg.eq.0) THEN
              IF (wcaenv(iwca2a).eq.'FALSE') THEN
                IF (trig_wca_stg(iwca2a).lt.regstg_wca(iwca2a,1)+
     &           0.50.and.(month.ge.12.or.month.le.5)) THEN
                  outf_to_wca2a=amin1(qoutflw(ia,i),s6max-kflo(ifls6))
                  qoutflw(ia,i)=outf_to_wca2a
                ELSE
                  qoutflw(ia,i)=0.0
                END IF
              ELSE
!          if (LWSLO(iwca2a) .or. LWSLO(iwca2b)) then
                IF (lwslo(iwca2a).or.(avg_sim_wca_stg(iwca2b)
     &           .lt.stage_import_wca(iwca2b)+offset(iwca2b,2)
     &           .and.avg_sim_wca_stg(iwca2b).lt.10.5)) THEN
                  outf_to_wca2a=amin1(qoutflw(ia,i),s6max-kflo(ifls6))
```

```
                  qoutflw(ia,i)=outf_to_wca2a
                ELSE
                  qoutflw(ia,i)=0.0
                END IF
              END IF
            ELSE
              qoutflw(ia,i)=0.0
            END IF
          ELSE IF (res_out_strname(ia,i).eq.'ST2T2B') THEN
!         node_299 = 30 - minx(35) + 1 + isum(35)
!         stage_299 = h(node_299) + pond(node_299)
!         if ( LRGLO(iwca2a) .or. stage_299 .gt. 10.7) then
            IF (lrglo(iwca2a).or.avg_sim_wca_stg(iwca2b)
     &       .gt.10.5.or.avg_sim_wca_stg(iwca2b)
     &       .gt.stage_import_wca(iwca2b)+offset(iwca2b,2)) THEN
              qoutflw(ia,i)=0.0
            END IF
          ELSE IF (res_out_strname(ia,i).eq.'STA5WO') THEN
            IF (16.0.gt.wcrsth) THEN
              IF (16.0 .gt.crstelev(ia,i,2).and.wcrsth .gt.
     &         crstelev(ia,i,2)) THEN
                crelev = crstelev(ia,i,2)
                submerg_fact = (wcrsth - crelev)/(16.0 - crelev)
                rel_crst_length = (16.0 - crelev)/(16.0 - crelev
     &          + 1.0)
                rmod_limit = 0.85 - 0.50*rel_crst_length
                reduc_fact = (submerg_fact -rmod_limit)
     &          /(1.0 -rmod_limit)
                reduc_fact = amax1(reduc_fact,0.0)
                reduc_fact = (1.0 - reduc_fact) ** 0.143
                qfactor=reduc_fact
              ELSE
                qfactor=1.0
              END IF
              qoutbkp=598.0*(16.0-crstelev(ia,i,2))**wrexp(ia
     &         ,i,2)*qfactor
            ELSE
              qoutbkp = 0.0
            ENDIF
            IF (head_av.gt.wcrsth) THEN
              IF (head_av .le. 16.0) then
                IF (head_av .gt.crstelev(ia,i,2)) THEN
                  crelev = crstelev(ia,i,2)
                  submerg_fact = (wcrsth - crelev)/(head_av - crelev)
                  rel_crst_length = (head_av - crelev)/(head_av - crelev
     &            + 1.0)
                  rmod_limit = 0.85 - 0.50*rel_crst_length
                  reduc_fact = (submerg_fact -rmod_limit)
     &            /(1.0 -rmod_limit)
                  reduc_fact = amax1(reduc_fact,0.0)
                  reduc_fact = (1.0 - reduc_fact) ** 0.143
                  qfactor=reduc_fact
                ELSE
                  qfactor=1.0
                END IF
                qoutflw(ia,i)=598.0*(head_av-crstelev(ia,i,2))**wrexp(ia
     &          ,i,2)*qfactor
```

```
              ELSE
                crelev = 16.0
                IF (head_av .gt.crelev) THEN
                  submerg_fact = (wcrsth - crelev)/(head_av - crelev)
                  rel_crst_length = (head_av - crelev)/(head_av - crelev
     &            + 1.0)
                  rmod_limit = 0.85 - 0.50*rel_crst_length
                  reduc_fact = (submerg_fact -rmod_limit)
     &            /(1.0 -rmod_limit)
                  reduc_fact = amax1(reduc_fact,0.0)
                  reduc_fact = (1.0 - reduc_fact) ** 0.143
                  qfactor=reduc_fact
                ELSE
                  qfactor=1.0
                END IF
                qoutflw(ia,i)=365.8*(head_av-16.0)**wrexp(ia,i,2)
     &          *qfactor + qoutbkp
              ENDIF
            ELSE
              qoutflw(ia,i) = 0.0
            ENDIF
          END IF
!      print *,QOUT,RESOUTMX,OUTLIM(IA,I),CHDEP(1),'  IN STAOUT'
!    $    ,HEAD,WCRSTH
!
          IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(19))
     &     THEN
!
!  determine if criteria are met for environmental ws releases
!   from LOK to WCA-3A
!
            ntarg=0
            DO itarg=1,no_targ_loc(is8bsn-6)
              IF (lwslo(itarg_area(itarg,is8bsn-6))) THEN
                ntarg=ntarg+1
              END IF
            END DO
          ELSE
            ntarg=0
          END IF
!
          IF ((lwslo(i_ds_target_indx(ia,i))
     &     .or.ntarg.gt.0.or.res_out_type(ia,i).eq.'FLDC  ')
     &     .or.opt_sim_code(ia,i).ne.'GEN') THEN
            IF (inodopt(ia,i,2).eq.1) THEN
!  outflow from grid cell
              IF (sfactor(ia) .GE. sfactmin .or. ires_small_sim
     &        (ia) .eq. 'NO ') THEN
!  reservoir is large enough not to treat as separate entity in
!  grid domain
                pond(idnstrm(ia,i))=pond(idnstrm(ia,i))-qoutflw(ia,i)*
     &          86400./(gdar*(1.0-carfact(idnstrm(ia,i))))
                IF (wrexp(ia,i,2) .NE. 1.0) res_struc_outflw(ia)
     &           = res_struc_outflw(ia) + qoutflw(ia,i)*1.9835
              ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
```

```
                DO ir = 1,noresincell(idnstrm(ia,i))
                 IF (ia .eq. ires_index_in_cell(idnstrm(ia,i),ir))
     &           THEN
                   IF (qoutflw(ia,i) .gt. 0.000001) THEN
                      stage_res(idnstrm(ia,i),ir)=stage_res
     &                (idnstrm(ia,i),ir) - amin1(qoutflw(ia,i)*86400.
     &                /(sfactor(ia)*gdar),pond_in_res(idnstrm(ia,i),ir))
     &                - amax1(qoutflw(ia,i)*86400./(sfactor(ia)*gdar)
     &                - pond_in_res(idnstrm(ia,i),ir),0.0)
     &                / s(idnstrm(ia,i))
                      pond_in_res(idnstrm(ia,i),ir)=pond_in_res(idnstrm
     &                (ia,i),ir) - amin1(qoutflw(ia,i)*86400./(sfactor
     &                (ia)*gdar),pond_in_res(idnstrm(ia,i),ir))
                   ELSE
                      sgn = -1.0
                      depth_to_wt = amax1(ells_in_res(idnstrm(ia,i),ir)
     &                - stage_res(idnstrm(ia,i),ir),0.0)
     &                *s(idnstrm(ia,i))
                      stage_res(idnstrm(ia,i),ir)=stage_res
     &                (idnstrm(ia,i),ir)+ amin1(sgn*qoutflw(ia,i)*86400.
     &                /(sfactor(ia)*gdar),depth_to_wt)/s(idnstrm(ia,i))+
     &                amax1(sgn*qoutflw(ia,i)*86400./(sfactor(ia)*gdar)
     &                - depth_to_wt,0.0)
                      pond_in_res(idnstrm(ia,i),ir) = amax1(stage_res
     &                (idnstrm(ia,i),ir) - ells_in_res(idnstrm(ia,i),ir)
     &                ,0.0)
                   ENDIF
                   IF (wrexp(ia,i,2) .EQ. 1.0) res_seep_out(ia)
     &              = res_seep_out(ia) + qoutflw(ia,i)*1.9835
c                  IF(resname(ia).eq.'R332C ') print *,qoutflw(ia,i)
c    &                ,'  R332CO  ',head_av,tail,month,iday
                   IF (wrexp(ia,i,2) .NE. 1.0) res_struc_outflw(ia)
     &              = res_struc_outflw(ia) + qoutflw(ia,i)*1.9835
                 ENDIF
                ENDDO
              ENDIF
            ELSE
!  outflow from canal
              qu(ioutcnl(ia,i))=qu(ioutcnl(ia,i))-qoutflw(ia,i)*86400.
            END IF
          ELSE
            qoutflw(ia,i)=0.0
          END IF
!
          IF (iwmads(ia,i).le.ntotres) THEN
            IF (inodopt(ia,i,3).eq.1) THEN
              DO j=1,ndspts(ia,i)
                IF (ibsn(nodepw(ia,i,j)).eq.ibsn_no_res(iwmads(ia,i)))
     &            THEN
                  volwtr(iwmads(ia,i))=volwtr(iwmads(ia,i))-qoutflw(ia,
     &             i)/ndspts(ia,i)
                END IF
              END DO
            ELSE
              DO j=1,ndspts(ia,i)
                IF (lcnb(ircnl(ia,i,j)).eq.ibsn_no_res(iwmads(ia,i)))
     &           THEN
```

```
                  volwtr(iwmads(ia,i))=volwtr(iwmads(ia,i))-qoutflw(ia,
     &             i)/ndspts(ia,i)
                END IF
              END DO
            END IF
          END IF
          qoutf=qoutflw(ia,i)
          jstr1=jstr1+1
          strflw(jstr1,idss)=qoutf
!
          IF (inodopt(ia,i,3).eq.0) THEN
            DO k=1,nwcas
              CALL match (cnm(ircnl(ia,i,1)), 1, 5, convey_canal_names,
     &         n_conv_tot(k), 5, imatch)
              IF (imatch.le.n_conv_tot(k)) THEN
                IF (k.eq.1) THEN
                  iconvey_canal_no=imatch
                ELSE
                  iconvey_canal_no=imatch-n_conv_tot(k-1)
                END IF
                i_wca=k
                GO TO 20
              END IF
            END DO
   20       IF (imatch.ne.0) THEN
              addflw(i_wca,iconvey_canal_no)=addflw(i_wca,
     &         iconvey_canal_no)+qoutf
            END IF
          END IF
          IF (resname(ia).eq.name_res_spec_code(8)) THEN
            out_to_wca1_frm_sta1w=qoutf
          END IF
!
          IF (resname(ia).eq.name_res_spec_code(6)) THEN
            out_to_wca1_frm_sta1e=qoutf
          END IF
!
          IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(26))
     &     THEN
            outf_from_c11res=qoutf
          END IF
          total_outflow_from_sta=total_outflow_from_sta+qoutf
!
!
          IF (opt_sim_code(ia,i).eq.'GEN') THEN
            IF (lwslo(i_ds_target_indx(ia,i))
     &       .or.ntarg.gt.0.or.res_out_type(ia,i).eq.'FLDC  ') THEN
!         IF (INODOPT(IA,I,2) .EQ. 1)
!    $    outf_fc_frm_res_to_cell(IDNSTRM(IA,I)) = 0.0
              IF ((sfactor(iwmads(ia,i)) .GE.sfactmin .or.ires_small_sim
     &        (iwmads(ia,i)) .eq. 'NO ') .OR. iwmads(ia,i) .GT. ntotres)
     &        THEN
              DO j=1,ndspts(ia,i)
                IF (inodopt(ia,i,3).eq.1) THEN
                  pond(nodepw(ia,i,j))=pond(nodepw(ia,i,j))+qoutf*
     &            86400./(ndspts(ia,i)*gdar*(1.0-total_small_afact
     &            (nodepw(ia,i,j))-carfact(nodepw(ia,i,j))))
```

```
              if (pond(nodepw(ia,i,j)) .lt. -0.0001) then
                  rchg(nodepw(ia,i,j)) = pond(nodepw(ia,i,j))
                  pond(nodepw(ia,i,j)) = 0.0
              endif
              IF (nodepw(ia,i,j).eq.idnstrm(ia,i) .AND.
&             (sfactor(ia) .GE. sfactmin .OR. ires_small_sim
&             (ia) .eq. 'NO ')) THEN
                IF (head_cell_beg(idnstrm(ia,i)).gt.ells(idnstrm(ia,
&                i))) THEN
                  IF (head.gt.stgmx(ia)+
&                  0.50.and.daily_outf_fc_frm_res_to_cell1(idnstrm
&                  (ia,i))
&                  .lt.daily_outf_from_res_to_cell_max(idnstrm(ia,i)
&                  )) THEN
                    IF (volwtr(ia).lt.res_area(ia)/(1.9835*1.5))
&                    THEN
                      outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                      outf_fc_frm_res_to_cell(idnstrm(ia,i))+
&                      daily_outf_from_res_to_cell_max(idnstrm(ia,i)
&                      )
                    ELSE
                      IF (weircff(ia,i,2).gt.80.0) THEN
                        outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                        outf_fc_frm_res_to_cell(idnstrm(ia,i))+
&                        daily_outf_fc_frm_res_to_cell1(idnstrm(ia,
&                        i))
                      ELSE
                        outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                        outf_fc_frm_res_to_cell(idnstrm(ia,i))+
&                        qoutf*86400./(ndspts(ia,i)*gdar)
                      END IF
                    END IF
                  ELSE
                    IF (weircff(ia,i,2).gt.80.0) THEN
                      outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                      outf_fc_frm_res_to_cell(idnstrm(ia,i))+
&                      daily_outf_fc_frm_res_to_cell1(idnstrm(ia,i))
                    ELSE
                      outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                      outf_fc_frm_res_to_cell(idnstrm(ia,i))+qoutf*
&                      86400./(ndspts(ia,i)*gdar)
                    END IF
                  END IF
                ELSE
                  IF (head.gt.ells(node)) THEN
                    outf_fc_frm_res_to_cell(idnstrm(ia,i))=
&                    outf_fc_frm_res_to_cell(idnstrm(ia,i))+qoutf*
&                    86400./(ndspts(ia,i)*gdar)
                  END IF
                END IF
                daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))=qoutf*
&                86400./(ndspts(ia,i)*gdar)
              END IF
            ELSE
              qu(ircnl(ia,i,j))=qu(ircnl(ia,i,j))+qoutf*86400./
&              ndspts(ia,i)
              rloc_excss_to_canl_next_step(ircnl(ia,i,j))=
```

```
     &             rloc_excss_to_canl_next_step(ircnl(ia,i,j))+qoutf*
     &             86400./ndspts(ia,i)
               END IF
               if (iwmads(ia,i) .le. ntotres)
     &         res_struc_inflw(iwmads(ia,i)) =res_struc_inflw
     &         (iwmads(ia,i)) + qoutf*1.9835/ndspts(ia,i)
             END DO
             IF (res_out_strname(ia,i).eq.'NLBS29') rnlkblt_to_s29=
     &        qoutf
             ELSE
               IF (inodopt(ia,i,3).eq.1) THEN
                  DO ir = 1,noresincell(nodepw(ia,i,1))
                   IF (iwmads(ia,i) .EQ. ires_index_in_cell
     &             (nodepw(ia,i,1),ir)) THEN
                     depth_to_wt = amax1(ells_in_res(nodepw(ia,i,1),ir)
     &               - stage_res(nodepw(ia,i,1),ir),0.0)*s(nodepw
     &               (ia,i,1))
                     stage_res(nodepw(ia,i,1),ir) = stage_res
     &               (nodepw(ia,i,1),ir) + amin1(qoutf*86400.
     &               /(ndspts(ia,i)*sfactor(iwmads(ia,i))*gdar)
     &               ,depth_to_wt)/s(nodepw(ia,i,1)) + amax1
     &               (qoutf*86400./(ndspts(ia,i)*sfactor(iwmads(ia,i))
     &               *gdar) - depth_to_wt,0.0)
                     pond_in_res(nodepw(ia,i,1),ir) = amax1(stage_res
     &               (nodepw(ia,i,1),ir) - ells_in_res(nodepw(ia,i,1)
     &               ,ir),0.0)
c                    if (resname(iwmads(ia,i)) .eq.'R332B ') print *,
c    &                 resname(ia),stage_res(nodepw(ia,i,1),ir)
c    &                 ,qoutf,'  R332BIN ',sfactor(iwmads(ia,i))
c    &                 ,sfactor(iwmads(ia,i))*gdar/43560.
                     res_struc_inflw(iwmads(ia,i)) =res_struc_inflw
     &               (iwmads(ia,i)) + qoutf*1.9835
                   ENDIF
                  ENDDO
               ENDIF
             ENDIF
           END IF
         ELSE
           IF (resname(ia).eq.name_res_spec_code(3)) THEN
             CALL bpts (iday, month, 10.0, 8, 1, 1, roten_sched)
             IF (rotenenv.eq.'TRUE ') THEN
               roten_import_stage=stage_import_wca(iroten)-
     &          offset(iroten,1)
             ELSE
               roten_import_stage=roten_sched+offset_to_sched(8)
             END IF
             canal_stage_roten=chdep(iroten_canal_no)+
     &        hdc(iroten_canal_no,iseas)
             canal_capac_roten=0.03*canal_stage_roten**4.127
             IF (roten_stage.le.roten_import_stage) THEN
               IF (iopt_for_semcyp_prior_roten.eq.1) THEN
                 outf_to_canal_sem=amin1(190.0,sem_dmnd_orig/1.9835,
     &            qoutf)
                 outflow_to_marsh=amin1(qoutf-outf_to_canal_sem,240.0)
               ELSE
                 outflow_to_marsh=amin1(qoutf,240.0)
               END IF
```

```
                rmarsh_above_capac=amax1(qoutf-outflow_to_marsh,0.0)
                outflow_to_canal_roten=amin1(rmarsh_above_capac,
     &           canal_capac_roten)
              ELSE
                outflow_to_canal_roten=amin1(qoutf,canal_capac_roten)
                outflow_to_marsh=amin1(qoutf-outflow_to_canal_roten,
     &           240.0)
              END IF
              qu(iroten_canal_no)=qu(iroten_canal_no)+
     &         outflow_to_canal_roten*86400./ndspts(ia,i)
              IF (inodopt(ia,i,3).eq.1) THEN
!           IF (INODOPT(IA,I,2) .EQ. 1)
!    $      outf_fc_frm_res_to_cell(IDNSTRM(IA,I)) = 0.0
                DO j=1,ndspts(ia,i)
                  pond(nodepw(ia,i,j))=pond(nodepw(ia,i,j))+
     &             outflow_to_marsh*86400./(ndspts(ia,i)*gdar
     &             * (1.0 - carfact(nodepw(ia,i,j))))
                  IF (nodepw(ia,i,j).eq.idnstrm(ia,i)) THEN
                    outf_fc_frm_res_to_cell(idnstrm(ia,i))=
     &               outf_fc_frm_res_to_cell(idnstrm(ia,i))+
     &               outflow_to_marsh*86400./(ndspts(ia,i)*gdar)
                    daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))=
     &               outflow_to_marsh*86400./(ndspts(ia,i)*gdar)
                  END IF
                END DO
              END IF
              qoutf=outflow_to_marsh+outflow_to_canal_roten
!        print *,canal_stage_roten,roten_stage,roten_import_stage
!    $ ,iroten_canal_no,canal_capac_roten,outflow_to_marsh
!    $ ,outflow_to_canal_roten,month,iday
            ELSE IF (resname(ia).eq.name_res_spec_code(4)) THEN
              outflow_from_sta6=qoutf
            ELSE IF (resname(ia).eq.name_res_spec_code(1)) THEN
              IF (res_out_strname(ia,i).eq.res_struc_name_spec_code(1)
     &         .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(2)
     &         .or.res_out_strname(ia,i).eq.res_struc_name_spec_code(3))
     &          THEN
                CALL match (name_res_spec_code(10), 1, 6, resname,
     &          ntotres+1, 6,ieaarsn_no)
                CALL match (name_res_spec_code(11), 1, 6, resname,
     &          ntotres+1, 6,ieaarss_no)
                IF (sta34_outf_flex_eaar_opt.eq.'YES') THEN
                  DO kk=1,nmtr
                    IF (ieaarsn_no.gt.0) THEN
                      IF (ibsn(mtrp(kk,1)).eq.ibsn_no_res(ieaarsn_no))
     &                 THEN
                       stg_eaa_res_2a=acstg(kk,1)
                       stg_eaar_min_2a = ells(mtrp(kk,1))+1.0
                      END IF
                    END IF
                    IF (ieaarss_no.gt.0) THEN
                      IF (ibsn(mtrp(kk,1)).eq.ibsn_no_res(ieaarss_no))
     &                 THEN
                       stg_eaa_res_2b=acstg(kk,1)
                       stg_eaar_min_2b=ells(mtrp(kk,1))+1.0
                      END IF
                    END IF
```

```
                  END DO
                END IF
                IF (lwslo(inewca3a).or.lwslo(iwca3a)
     &          .or.(lwslo_eaares(inewca3a)
     &          .and.stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0.and.sta34_outf_flex_eaar_opt.eq.'YES')
     &          .or.(lwslo_eaares(inewca3a).and.
     &          stg_eaa_res_2b.ge.stg_eaar_min_2b.and.ieaarss_no.gt.0
     &          .and.sta34_outf_flex_eaar_opt.eq.'YES')
     &          .or.opt_outflow_from_res_to_wca.eq.'FLDC') THEN
                  DO j=1,ndspts(ia,i)
                    IF (inodopt(ia,i,3).eq.1) THEN
                      pond(nodepw(ia,i,j))=pond(nodepw(ia,i,j))+qoutf*
     &                 86400./(ndspts(ia,i)*gdar*(1.0-total_small_afact
     &                 (nodepw(ia,i,j))-carfact(nodepw(ia,i,j))))
                    ELSE
                      qu(ircnl(ia,i,j))=qu(ircnl(ia,i,j))+qoutf*86400./
     &                 ndspts(ia,i)
                      rloc_excss_to_canl_next_step(ircnl(ia,i,j))=
     &                 rloc_excss_to_canl_next_step(ircnl(ia,i,j))+
     &                 qoutf*86400./ndspts(ia,i)
                    END IF
                  END DO
                  sta34_out_to_wca3ane=qoutf+sta34_out_to_wca3ane
                ELSE
                  pot_exc_outf_frm_sta34_to_nca3a=qoutf+
     &             pot_exc_outf_frm_sta34_to_nca3a
                END IF
c               print *,sta34_outf_flex_eaar_opt
c    &           ,opt_outflow_from_res_to_wca
c    &           ,sta34_out_to_wca3ane,pot_exc_outf_frm_sta34_to_nca3a
c    &           ,qoutf,'  ST3OT'
              ELSE IF (res_out_strname(ia,i)
     &         .eq.res_struc_name_spec_code(8)) THEN
                DO j=1,ndspts(ia,i)
                  IF (inodopt(ia,i,3).eq.1) THEN
                    pond(nodepw(ia,i,j))=pond(nodepw(ia,i,j))+qoutf*
     &               86400./(ndspts(ia,i)*gdar*(1.0-carfact
     &               (nodepw(ia,i,j))-total_small_afact
     &               (nodepw(ia,i,j))))
                  ELSE
                    qu(ircnl(ia,i,j))=qu(ircnl(ia,i,j))+qoutf*86400./
     &               ndspts(ia,i)
                  END IF
                END DO
              END IF
            ELSEIF (resname(iwmads(ia,i)) .eq. name_res_spec_code(6))
     &      THEN
              node_sta1ee = 32 - minx(53) + 1 + isum(53)
              pond(nodepw(ia,i,1)) = pond(nodepw(ia,i,1))
     &        + qoutf*86400.*frac_s319_sta1ew / gdar
              pond(node_sta1ee) = pond(node_sta1ee)
     &        + qoutf*86400.* frac_s319_sta1ee / gdar
            ELSE
              IF (resname(ia).ne.name_res_spec_code(5).and.resname(ia)
     &         .ne.name_res_spec_code(4)) THEN
!         IF (INODOPT(IA,I,2) .EQ. 1)
```

```
!    $   outf_fc_frm_res_to_cell(IDNSTRM(IA,I)) = 0.0
                DO 30 j=1,ndspts(ia,i)
                  IF (inodopt(ia,i,3).eq.1) THEN
                    pond(nodepw(ia,i,j))=pond(nodepw(ia,i,j))+qoutf*
     &               86400./(ndspts(ia,i)*gdar*(1.0-total_small_afact
     &               (nodepw(ia,i,j))-carfact(nodepw(ia,i,j))))
                    IF (nodepw(ia,i,j).eq.idnstrm(ia,i)) THEN
                      outf_fc_frm_res_to_cell(idnstrm(ia,i))=
     &                 outf_fc_frm_res_to_cell(idnstrm(ia,i))+qoutf*
     &                 86400./(ndspts(ia,i)*gdar)
                      daily_outf_fc_frm_res_to_cell(idnstrm(ia,i))=
     &                 qoutf*86400./(ndspts(ia,i)*gdar)
                    END IF
                  ELSE
                    qu(ircnl(ia,i,j))=qu(ircnl(ia,i,j))+qoutf*86400./
     &               ndspts(ia,i)
                    rloc_excss_to_canl_next_step(ircnl(ia,i,j))=
     &               rloc_excss_to_canl_next_step(ircnl(ia,i,j))+qoutf*
     &               86400./ndspts(ia,i)
                  END IF
   30           CONTINUE
              END IF
            END IF
          END IF
   40   CONTINUE
!
!   calculate hydraulic detention time for reservoirs
!
        IF (total_outflow_from_sta.gt.0.0) THEN
          hydraulic_det_time(ia)=(res_area(ia)*mean_ponding_depth(ia)*
     &     43560.)/(total_outflow_from_sta*86400.)
          hydraulic_det_time(ia)=amin1(hydraulic_det_time(ia),199.0)
        ELSE
          hydraulic_det_time(ia)=199.0
        END IF
   50 CONTINUE
!
!   route total outflow from rotenberger to appropriate areas
!
      IF (runoff_to_holeyland.eq.'INDIRCT') THEN
        outflow_from_roten_to_holy=amin1(total_outflow_from_roten,
     &   pumpcap_res(1))
        rem_inflow_to_meet_holy_sched=pumpcap_res(1)-
     &   outflow_from_roten_to_holy
        total_outflow_from_roten=total_outflow_from_roten-
     &   outflow_from_roten_to_holy
        outflow_from_roten_to_west=amin1(total_outflow_from_roten,
     &   capc_new_pump_to_west_frm_roten)
        outflow_from_roten_thru_s8=total_outflow_from_roten-
     &   outflow_from_roten_to_west
        qu(int_cnl_holey(1))=qu(int_cnl_holey(1))+
     &   outflow_from_roten_to_holy*86400.
      ELSE
        outflow_from_roten_thru_s8=total_outflow_from_roten-
     &   outflow_from_roten_to_west
      END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0--
```

```
      DO 250 I = MAXY,1,-1
         DO 240 NODE = ISUM(I) +  1, ISUM(I+1)
c
c  incorporate small reservoirs as part of grid system
c
            total_sfactor = 0.0
            GWHEADnode = 0.0
            DO ir = 1,noresincell(node)
               ia = ires_index_in_cell(node,ir)
               IF (sfactor(ia) .LT. sfactmin .AND. ires_small_sim
     $         (ia) .eq. 'YES') THEN
                  total_sfactor = sfactor(ia) + total_sfactor
                  GWHEADnode = stage_res(node,ir)*sfactor(ia)
     $            + GWHEADnode
                  cell_storage(node) = cell_storage(node)
     $            + (amax1(stage_res(node,ir),ells_in_res(node,ir))
     $            - ells_in_res(node,ir) + (amin1(stage_res(node,ir)
     $            ,ells_in_res(node,ir)) - ref_stg_for_cell_stor)
     $            * s(node)) * sfactor(ia) * gdar/43560.
     $
               endif
            enddo
            DO ic = 1,no_canl_in_cell(node)
              kcnl_no = icanl_no_in_cell(node,ic)
              if (nodcr(kcnl_no) .gt. 1) then
                stgdiff_canal_ups=float(nodcr(kcnl_no)-
     $          ipos_cell_in_canl(node,ic))/float(nodcr(kcnl_no)-1)
     $          *hdc(kcnl_no,iseas)
              else
                stgdiff_canal_ups=0.0
              endif
              canl_stage_in_cell = chdep(kcnl_no) + stgdiff_canal_ups
              canl_stor_in_cell(node) = canl_stor_in_cell(node)
     $        + (canl_stage_in_cell+10) * rindiv_canl_area(node,ic)
     $        / 43560.
            ENDDO
            GWHEADnode1 = GWHEADnode + (H(NODE)+POND(NODE)+solmx(node)
     $      /s(node))*(1.0 - total_sfactor - carfact(node))
            cell_storage(node) = cell_storage(node) +
     $      ((H(NODE)-ref_stg_for_cell_stor) * S(NODE)+solmx(node)
     $      +POND(NODE))*(1.0 - total_sfactor - carfact(node))
     $      *GDAR / 43560.
            total_cell_stor(node) = cell_storage(node)
     $      + canl_stor_in_cell(node)
            STAGEND(NODE) = GWHEADnode1
  240    CONTINUE
  250 CONTINUE
      RETURN
      END
```

resws.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: resws.inc,v 1.6 2004/03/05 21:20:28 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/resws.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /resws/ noutlets(max_n_res),
     &idshg_opt(max_n_res,max_nostr_res),
     &istrctr_ws(max_n_res,max_nostr_res),
     &ieaa_bsn(max_n_res,max_nostr_res),nres_ws,
     &stg_min_wsply(max_n_res,max_nostr_res),
     &iconv_use_opt_ws(max_n_res),n_cnls_dn_ws(max_n_res,max_nostr_res),
     &wsndese_rev,pot_exc_outf_frm_sta34_to_nca3a,wsndes_rev,
     &no_gravity_outlets_sta34,igrav_outlet_no(max_nostr_res),
     &ws_down_sta2,ratio_rem_sta2,ws_frm_res_to_wca1,
     &ws_frm_res_to_wca2a,ws_frm_res_to_wca3a,
     &i_ds_target_indx(max_n_res,max_nostr_res),
     &opt_sim_code(max_n_res,max_nostr_res),n_conv_tot(max_n_wcas),
     &env_ws_from_res_thru_g372,indx_sta1e_frm_lok_ws,
     &wsdel_to_sta(max_n_res),ws_to_sta5,ws_to_sta6,
     &indx_sta1w_frm_lok_ws,indx_sta1ee_frm_lok_ws,
     &indx_sta2w_frm_lok_ws
C-----------------------------------
      CHARACTER opt_sim_code*3
```

reuse.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: reuse.inc,v 1.5 2003/07/03 20:08:04 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/reuse.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /reuse/ no_of_reuse_plnts,
     &no_canals_reuse(max_n_reuse_plnts),
     &node_reuse(max_n_reuse_plnts,max_n_cells_reuse),
     &avg_daily_reuse_vol(max_n_reuse_plnts,12),
     &no_grid_cells_reuse(max_n_reuse_plnts),
     &icanal_reuse_indx(max_n_reuse_plnts,max_n_cnls_reuse),
     &iopt_rec_reuse(max_n_reuse_plnts),
     &total_reuse_res,ires_no_reuse(max_n_reuse_plnts,max_n_res_reuse),
     &opt_time_var_reuse
C-------------------------------------------

      character opt_time_var_reuse*5
```

reusenores.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: reusenores.F,v 1.9 2005/06/10 13:29:19 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/reusenores.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine routes reuse water to appropriate location(s).
C   Reuse water can be routed to a canal or to grid cells which
C   may contain reservoir(s).
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE reusenores (total_reuse, dqu, upstream_inflow,
     & total_reuse_ws, resname_reuse, total_afact, sumrain,
     & rmax_stage_reuse, stage_res)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'reuse.inc'

      DIMENSION avail_vol_reuse(max_n_reuse_plnts,max_n_cells_reuse),
     & dqu(max_ncnls), resname_reuse(max_n_reuse_plnts,max_n_res_reuse),
     & rmax_stage_reuse(max_n_reuse_plnts,max_n_cells_reuse),
     & stage_res(max_ncells,max_n_res_in_cell),
     & total_afact(max_ncells),sumrain(max_ncells),
     & total_reuse_ws(max_n_serv_area), upstream_inflow(max_ncnls)

      CHARACTER resname_reuse*6

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  add water reuse to appropriate locations
```

```
      total_reuse = 0.0
      DO i = 1,no_of_reuse_plnts
         IF (iopt_rec_reuse(i) .EQ. 0) THEN

C  reuse to canals
            DO k = 1,no_canals_reuse(i)
               qu(icanal_reuse_indx(i,k)) = qu(icanal_reuse_indx(i,k)) +
     $         avg_daily_reuse_vol(i,month)*43560. / no_canals_reuse(i)
               upstream_inflow(icanal_reuse_indx(i,k)) = upstream_inflow
     $         (icanal_reuse_indx(i,k)) + avg_daily_reuse_vol(i,month)
     $         * 43560. / no_canals_reuse(i)
               dqu(icanal_reuse_indx(i,k)) = dqu(icanal_reuse_indx(i,k))
     $         - amin1(avg_daily_reuse_vol(i,month) / 1.9835, dqu
     $         (icanal_reuse_indx(i,k)))

               IF (iserv_area_loc(icanal_reuse_indx(i,k)) .GT. 0) THEN
                  total_reuse_ws(iserv_area_loc(icanal_reuse_indx(i,k)))
     $            = total_reuse_ws(iserv_area_loc(icanal_reuse_indx(i,k)
     $            )) + amin1(avg_daily_reuse_vol(i,month) / 1.9835,dqu
     $            (icanal_reuse_indx(i,k)))
               ENDIF
               ICNL_DNS = icanal_reuse_indx(i,k)
   10          ICNL_DNS = iup_canal_ws(ICNL_DNS)
               IF (ICNL_DNS .LE. NCH) THEN
                  dqu(ICNL_DNS) = dqu(ICNL_DNS) - avg_daily_reuse_vol
     $            (i,month) / 1.9835
                  dqu(ICNL_DNS) = AMAX1(dqu(ICNL_DNS),0.0)
                  GO TO 10
               ENDIF
            END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            total_reuse = total_reuse + avg_daily_reuse_vol(i,month)
     $      / 1.9835
         ELSE
c  reuse to grid cell location(s)
            DO k=1,no_grid_cells_reuse(i)
               IF (resname_reuse(i,k) .EQ. 'NORES ') THEN
                  pond(node_reuse(i,k)) = pond(node_reuse(i,k)) +
     $            avg_daily_reuse_vol(i,month) * 43560. /(gdar
     $            *no_grid_cells_reuse(i))
                  total_reuse = total_reuse + avg_daily_reuse_vol
     $            (i,month) / (1.9835 * no_grid_cells_reuse(i))
               ELSE
                  ia1=ires_no_reuse(i,k)
                  node=node_reuse(i,k)
                  DO ir = 1,noresincell(node)
                  IF(ia1 .eq. ires_index_in_cell(node,ir)) THEN
                    depth_to_wt = (ells(node)-h(node))*s(node)
                    if (pond(node) .gt. depth_to_wt) THEN
                      stage_reuse=pond(node)-depth_to_wt
     $                +ells(node)
                    else
                      stage_reuse=(pond(node)-depth_to_wt)/s(node)
     $                + ells(node)
                    endif
                    IF (stage_reuse.le.ells(node)) THEN
```

```
                 afact=1.0
                ELSE
                 afact=total_afact(node)
                END IF
                adjust_for_ovlf=sum_chg_dpth_ovl_flw(node)
                term1_for_adjust_area=(sumet_outside_res(node)
&               +seep_loss_res(node)-sumrain(node)/12.0
&               -sum_chg_depth_levee_spg(node))
&               *((1.0-afact)/afact)
                term2_for_adjust_area=(outf_fc_frm_res_to_cell(node)
&               +outf_seep_loc_to_res_cell(node)+adjust_for_ovlf)
&               /afact
                IF (stage_reuse.gt.ells(node_reuse(i,k))) THEN
                  depsw=(stage_reuse-ells(node))/afact+
&                 term1_for_adjust_area-adjust_dur_excess(node)-
&                 term2_for_adjust_area
                  depsw=depsw-adjust_for_area_m1(node)
                  stage_res(node,ir)=ells(node)+depsw
                  IF (rmax_stage_reuse(i,k).ge.ells(node)) THEN
                    avail_vol_reuse(i,k)=(rmax_stage_reuse(i,k)-
&                   stage_res(node,ir))*sfactor(ia1)*gdar/43560.
                  ELSE
                    avail_vol_reuse(i,k)=0.0
                  END IF
                 ELSE
                   avail_vol_reuse(i,k)=amax1(rmax_stage_reuse(i,k)-
&                  ells(node),0.0)*sfactor(ia1)*gdar/43560.+
&                  (amin1(ells(node),rmax_stage_reuse(i,k))
&                  -stage_reuse)*s(node)*gdar/43560.
                 END IF
               END IF
               END DO
               avail_vol_reuse(i,k)=amax1(avail_vol_reuse(i,k),0.0)
               pond(node)=pond(node)+amin1(avail_vol_reuse(i,k),
&              avg_daily_reuse_vol(i,month)/no_grid_cells_reuse(i))*
&              43560./gdar
               total_reuse_res=total_reuse_res+
&               amin1(avg_daily_reuse_vol(i,month)/
&               no_grid_cells_reuse(i),avail_vol_reuse(i,k))/1.9835
            ENDIF
        END DO
    ENDIF
 END DO
 RETURN
 END
```

routc.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C     CVS Keywords
C     $Author: wwilcox $
C     $Id: routc.inc,v 1.5 2005/06/10 13:29:24 wwilcox Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/routc.inc,v $
C       $Name: rel-5-5 $
-
C                                                                       -
C                                                                       -
C-----------------------------------------------------
      common /ROUTC/ NODES9,LS90N,NODES13,L31W,S351CAP,
     &iroute_opt(max_n_str_node_dep),IFLS357,IG31W,LC1W,
     &cnlcap_ws(max_n_eaa_condt),HOLPUMP,
     &iup_canal_index(max_n_str_lok_in_wca),NODS151,IFLG54,IFLS8,IFLS7,
     &IS140UP,LRESON,idn_canal_index(max_n_str_lok_in_wca),IFLS6,IFLS5A,
     &IFLS77,IFLS308,IS8BSN,IS7BSN,IS6BSN,IS5ABSN,iseas,IFLHG5,IFLBEL5,
     &IGL38,IFLS2,IFLS3,IG2431,node_hw(max_n_str_lok_in_wca),
     &expon(max_n_str_lok_in_wca),IG596,IFLS5A5,IFLL8,ISA1,ISA2,ICA1,
     &ICA2,ICA3,ISA3,NODE_ROTEN1,LC111E,IRESID,QRNOFFM,QWTSPM,NODEHL,
     &IC18,IC12,node_tw(max_n_str_lok_in_wca),ifls332,QHLWSM,IFLAGQ,
     &ILAGBSN,FRF332,IFLS140,NODS140,s140cap,
     &ihw_opt(max_n_str_lok_in_wca),QS7FACT,QS6FACT,QS150FT,IMCNL,ILKMG,
     &IFLS150,IFLL8WS,IG2517,IFLS2MC,IHILL,INNR,IWPB,IMIAMI,S351MX,
     &S352MX,S354MX,IFLS32A,CNLCAP(max_n_eaa_condt),IS339UP,IL38,
     &RMAXSTG(max_n_eaa_condt),s8_cap,s7_cap,S8MAX,IL8W,S7MAX,S6MAX,
     &S5AMAX,S2MAX,S3MAX,CRRNFF(max_n_eaa_condt),
     &COEFR(max_n_eaa_condt,2),S354CAP,LS151,IWCA1,IWCA2A,IWCA3A,IWCA3B,
     &IWCA2B,TARGNOTMT,NODCAT,is150bws,IMAKEUP,BMPVOL,TARGET,G136EAA,
     &FLWS319,IG136,NODL101,nmdays,IFLS316,ICTSTA(max_n_str_lok_in_wca),
     &REGS354,REGS351,IFLS309,REGS352,ndays_sim,IFLC10A,IL8,IC51,TOTF,
     &IG155,IG89,IG88,IC51W,IL8S,itw_opt(max_n_str_lok_in_wca),S5A4MAX,
     &IFLG89,CAPACCS2,CAPACCS9,CAPACCS12,CAPACCS17W,iloxah,CAPACCS17E,
     &IFLCS2,IFLCS4,IFLCS9,IFLCS12,IFLCS17W,IFLCS17E,S352CAP,s8_cap_new,
     &s7_cap_new,flow_lower_reach_max,iholy,iroten,lc111,igravity_s8,
     &igravity_s7,flow_to_import_wpb,IGS357DN,CAPACCS4,
     &dischg_c(max_n_str_lok_in_wca),g136_to_sta34,rmiami_runoff_to_sta,
     &rnnrc_runoff_to_sta,ig200a,eaa_runff_s5a_to_sta,
     &struc_name_spec_code(max_n_str_lok_in_wca),
     &rmax_stg_s12s343(5,12),no_of_struc_w_ds_targ_no_res,
```

```
     &il8bsn,frac_capac_s343ab(12),adjust_for_area_m2(max_ncells),
     &adjust_for_area_m3(max_ncells),LFORPUMP(max_n_reach_eaa),
     &adjust_for_area_m4(max_ncells),adjust_for_area_m(max_ncells),
     &forw_pump_cap(max_n_eaa_condt)
C-----------------
      LOGICAL HOLPUMP,LRESON,LS151,LS90N,LFORPUMP
      character struc_name_spec_code*6
```

simcstg2dss.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: simcstg2dss.F,v 1.6 2003/07/03 20:07:45 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/simcstg2dss.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE DUMPS DAILY FLOWS FROM SIMULATED STRUCTURE INTO
C   THE DSS FILE 'STRFLOWS.DSS'; IF PATHNAMES ALREADY EXIST, IT WRITES
C   OVER THEM; IF NOT, IT APPENDS THESE SIMULATED FLOWS AT THE END OF
C   THE DSS FILE
C
C  THIS SUBROUTINE APPENDS TO FILE str2x2.dss SIMULATED FLOWS
C      FROM THE SFWMM.   THE TYPICAL DSS PATHNAME FOR A
C      STRUCTURE HAS THE FOLLOWING A THROUGH F PARTS:
C
C         A       B     C  D  E      F
C      /SFWMM/cnlname/STAGE//1DAY/SIMULATED/
C
C  INTEGER ARRAY STORING DSS INITIALIZATION TABLES
C     CHARACTER DSSFN*40,SDATE*9,STIME*4
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE simcstg2dss (ifltab, sdate, jbm)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'simcstgs.inc'
      INCLUDE 'simflows.inc'

      DIMENSION tmp(366)

      CHARACTER*8 ctype, cunits
```

```
      CHARACTER*80 ca, cb, cc, cd, ce, cf, cin, cpath
      CHARACTER sdate*9, stime*4

      INTEGER ifltab(1200)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF(JBM.EQ.IBM)GOTO 30
      STIME='1200'

C  FOR EACH DISCHARGE SET, FORM PATHNAME AND WRITE TO DSS

      CUNITS='FT MSL'
      CTYPE='PER-AVER'
      CA='SFWMM'
      NA=5
      CC='STAGE'
      NC=5
      CD=SDATE
      ND=9
      CE='1DAY'
      NE=4
      CF='SIMULATED'
      NF=9
      IPLAN=0

C FORM PATHNAME FROM COMPONENT NAMES AND STORE INTO DSS FILE
C NOTE: WHAT ARE BEING PASSED ARE DAILY AVERAGE FLOWS IN CFS
C
      DO 20 I = 1,MAXSTRC
         CIN=CNLNAME(I)
         CALL REMBLK(CIN,CB,NB)
         IF (NB.EQ.0) THEN
            WRITE(*,*) I, 'Canal name is missing!'
            STOP
         ENDIF
C
         DO 10 J = 1,IDSS
            TMP(J) = CNLSTAGE(I,J)
   10    CONTINUE
C
         CALL ZFPN (CA,NA,CB,NB,CC,NC,' ',0,CE,NE,CF,NF,CPATH,NPATH)
         CALL ZSRTS (IFLTAB,CPATH,SDATE,STIME,IDSS,TMP,CUNITS,CTYPE,
     $   IPLAN,ISTAT)
   20 CONTINUE
C
C
C INITIALIZE STRUCTURE FLOW ARRAY
C
   30 DO 50 I=1,MAXSTRC
         DO 40 J=1,366
            CNLSTAGE(I,J) = 0.0
   40    CONTINUE
   50 CONTINUE
C
      RETURN
      END
```

simcstgs.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: simcstgs.inc,v 1.3 2003/07/03 20:08:05 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/simcstgs.inc,v $
C       $name$                                                          -
C                                                                       -
C                                                                       -
C-----------------------------------------------------
      CHARACTER CNLNAME(200)*5
      common /SIMCNL/ CNLNAME
      common /SIMCSTG/ JSTRC,CNLSTAGE(max_ncnls,366),MAXSTRC
```

simflows.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: simflows.inc,v 1.3 2003/07/03 20:08:05 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/simflows.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      CHARACTER STRNAME(800)*6,sname_env_dss(30,5)*6
     $ ,sname_reg_dss(30,5)*6,sname_ws_dss(30,5)*6
      common /SIMSTR/ STRNAME,sname_env_dss,sname_reg_dss,sname_ws_dss
      common /SIMFLOWS/ IDSS,MAXSTR,JSTR,
     &STRFLW(max_n_entries_sim_flw_outp,366),JSTR1
```

simq2dss.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: simq2dss.F,v 1.6 2003/07/03 20:07:45 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/simq2dss.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE DUMPS DAILY FLOWS FROM SIMULATED STRUCTURE INTO
C   THE DSS FILE 'STRFLOWS.DSS'; IF PATHNAMES ALREADY EXIST, IT WRITES
C   OVER THEM; IF NOT, IT APPENDS THESE SIMULATED FLOWS AT THE END OF
C   THE DSS FILE
C
C  THIS SUBROUTINE APPENDS TO FILE str2x2.dss SIMULATED FLOWS
C      FROM THE SFWMM.   THE TYPICAL DSS PATHNAME FOR A
C      STRUCTURE HAS THE FOLLOWING A THROUGH F PARTS:
C
C         A       B     C  D  E      F
C      /SFWMM/strname/FLOW//1DAY/HISTORICAL/
C
C  INTEGER ARRAY STORING DSS INITIALIZATION TABLES
C     CHARACTER DSSFN*40,SDATE*9,STIME*4
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE simq2dss (ifltab, sdate, jbm)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'simflows.inc'

      DIMENSION tmp(366)

      CHARACTER*8 ctype, cunits
      CHARACTER*80 ca, cb, cc, cd, ce, cf, cin, cpath
      CHARACTER sdate*9, stime*4
```

```
      INTEGER ifltab(1200)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF(JBM.EQ.IBM)GOTO 30
      STIME='1200'

C  FOR EACH DISCHARGE SET, FORM PATHNAME AND WRITE TO DSS

      CUNITS='CFS'
      CTYPE='PER-AVER'
      CA='SFWMM'
      NA=5
      CC='FLOW'
      NC=4
      CD=SDATE
      ND=9
      CE='1DAY'
      NE=4
      CF='SIMULATED'
      NF=9
      IPLAN=0
C23456789*123456789*123456789*123456789*123456789*123456789*123456789*12
C
C FORM PATHNAME FROM COMPONENT NAMES AND STORE INTO DSS FILE
C NOTE: WHAT ARE BEING PASSED ARE DAILY AVERAGE FLOWS IN CFS
C
      DO 20 I = 1,MAXSTR
         CIN=STRNAME(I)
         CALL REMBLK(CIN,CB,NB)
         IF (NB .EQ. 0) THEN
            WRITE(*,*) I, 'Structure name is missing!'
            STOP
         ENDIF
C
         DO 10 J = 1,IDSS
            TMP(J) = STRFLW(I,J)
   10    CONTINUE
C
         CALL ZFPN (CA,NA,CB,NB,CC,NC,' ',0,CE,NE,CF,NF,CPATH,NPATH)
         CALL ZSRTS (IFLTAB,CPATH,SDATE,STIME,IDSS,TMP,CUNITS,CTYPE,
     $   IPLAN,ISTAT)
   20 CONTINUE
C
C
C INITIALIZE STRUCTURE FLOW ARRAY
C
   30 DO 50 I=1,MAXSTR
         DO 40 J=1,366
            STRFLW(I,J) = 0.0
   40    CONTINUE
   50 CONTINUE
C
      RETURN
      END
```

sl.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: sl.F,v 1.11 2005/06/10 13:29:19 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/sl.F,v $
C   $Name: rel-5-5 $
C
C
C  definition of terms: (all units in ac-ft/day)
C  input:
C        c44dmd      = c44 basin demand
C        c44ro       = c44 basin runoff
C        sledmd      = caloosahatchee estuarine demand
C        cbslest     = fraction of C44 estuarine demand to be met
C                      (based on ssm, for now)
C        cbc44       = fraction of c44 basin demand to be met
C                      as calculated in the supply-side management
C                      SUBROUTINE
C        stglok      = lake okeechobee stage
C
C  output:
C        bsnsto      = portion of c44 basin runoff that has to stay as
C                      storage in the basin since outlet capacities
C                      are already exceeded
C        unmetdmd    = portion of c44 basin demand that cannot be met
C                      by LOK
C
C        ealc44ro    = c44 basin runoff used to meet estuarine demand
C        eallok1     = discharge from LOK used to meet estuarine demand
C        rmin_lok_stg_del_est = minimum LOK stage allowed to meet
C                               estuarine demand
C        alc44       = c44 basin allocation;
C                      c44 basin demand multiplied by the cutback cbc44
C        eflc44      = c44 basin runoff in excess of estruarine demand
C                      which can go to LOK or the estuarine
C        eallok2     = discharge from LOK used to meet c44 basin demand
C        eqs80       = total flow thorugh s80 (excludes regulatory flows)
```

```
C        eqs308      = total flow thorugh s308 (excludes regulatory flows)
C
C-------------------------------------------------------------------------------
C    This subroutine determines inflows/outflows between Lake Okeechobee,
C    St. Lucie basin,and St. Lucie estuary. This requires daily time
C    series of basin demand/runoff and a daily time series of estuarine
C    demands.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE stlucie1 (iyr, imo, ida, c44dmd, c44ro, sltinf, sledmd,
     & cbslest, cbc44, stglok, bsnsto, unmetdmd, dmd_met_by_lok, stgref,
     & rmulti_seas_pred, clim_threshold_est)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'cstlucie.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'opersched.inc'

      DIMENSION clim_threshold_est(2), rmulti_seas_pred(12)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C initialize allocations and estimated deliveries

      alsle = 0.0
      ealc44ro = 0.0
      eallok1 = 0.0
      sledmd_met_by_lok = 0.0
      c44dmd_met_by_lokres = 0.0
      alc44 = 0.0
      eflc44 = 0.0
      eallok2 = 0.0
C     eqs80 = 0.0
C     eqs308 = 0.0
      dmd_met_by_lok = 0.0
      unmetdmd = 0.0
      excess_ro_to_est = 0.0
      excess_ro_to_lok = 0.0
      excess_ro_to_sto = 0.0
      c44ro0 = c44ro
c
c determine zone of schedule in which LOK stage resides
c
      CALL bpts(ida,imo,stglok,1,mzone,0,stgref_est)
      CALL bpts(ida,imo,10.0,1,n_fc_zones+6, 1,
     $rmin_lok_stg_del_est_normf)
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c adjust st lucie estuary demand by st lucie tributary inflow
c note: allocation will be based on sledmd adjusted for inflows
c       from other tributaries to estuary
      IF (sltinf .GT. sledmd)THEN
         est_dmd_met_by_local_ro = sledmd
         excess_local_ro = sltinf - sledmd
      ELSE
         est_dmd_met_by_local_ro = sltinf
```

```
         excess_local_ro = 0.0
      ENDIF
      sledmd = amax1(0.0,sledmd-sltinf)
c
c calculate st lucie estuary allocation
c
      alsle = cbslest*sledmd
      alsle0 = alsle
      IF (alsle .EQ. 0.0) goto 10
c estimate portion of allocation to be met by c44 basin runoff
      ealc44ro = amin1( alsle, c44ro)
      IF (ealc44ro .EQ. alsle) THEN
         c44ro = c44ro - ealc44ro
         alsle = 0.0
      ELSE
         alsle = alsle - c44ro
         c44ro = 0.0
      ENDIF
c update estimated structure flow
      eqs80 = eqs80 + ealc44ro
      s80cap = s80cap - ealc44ro
c estimate portion of allocation to be met by Lake Okeechobee
c assumption: LOK is always capable to meet remaining estuarine demand
      IF ((stglok .GT. rmin_lok_stg_del_est .AND. iclimate_opt .EQ. 0)
     $.OR. (stglok .GT. rmin_lok_stg_del_est .AND. ((rmulti_seas_pred
     $(imo) .GT. clim_threshold_est(1) .AND. stglok .GE.
     $rmin_lok_stg_del_est_normf .AND. mzone .GE. mzone_puls_top) .OR.
     $(rmulti_seas_pred(imo) .GT. clim_threshold_est(2) .AND. stglok
     $.LT. rmin_lok_stg_del_est_normf) .OR. (mzone .LT. mzone_puls_top
     $.AND. no_puls_zones .GT .0) .OR. (mzone .LE. n_fc_zones .AND.
     $no_puls_zones .EQ. 0)))) THEN

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         eallok1 = alsle
         eallokres2 = 0.0
      ELSE
         eallok1 = 0.0
         IF (lok_split_option .EQ. 1 .AND. stagelo_res .GE. 2.0) THEN
            eallokres2 = alsle
         ELSE
            eallokres2 = 0.0
         ENDIF
      ENDIF
c update estimated structure flows
      eqs308 = eqs308 + eallok1 + eallokres2
      s308cap = s308cap - eallok1 - eallokres2
      eqs80 = eqs80 + eallok1 + eallokres2
      s80cap = s80cap - eallok1 - eallokres2
      sledmd_met_by_lok = eallok1 + eallokres2
c
c excess C-44 BASIN runoff accounting
c
c determine where excess flow from c44 basin will go after
c estuary's allocations have been met
   10 IF (c44ro .NE. 0.0) THEN
         eflc44 = c44ro
         IF (lok_split_option .EQ. 1) THEN
```

```
            stglokup = stagelo_res
            stg_ref  = 14.5
         ELSE
            stglokup = stglok
            CALL bpts(ida,imo,10.0,1,n_fc_zones,1,stg_ref)
         ENDIF
         CALL bpts(ida,imo,10.0,1,n_fc_zones + 8,1,stg_backflw_stl)
         stg_backflow = stg_backflw_stl
         IF (stglokup.GT.stg_backflow) THEN
c LOK stage too high, release to tide
            excess_ro_to_est = eflc44
            eqs80 = eqs80 + eflc44
            s80cap = s80cap - eflc44
            IF (s80cap .LT. 0.0) THEN
c capacity of s80 exceeded, portion of excess runoff stays in C-44 basin
c put back flows
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               s80cap = s80cap + eflc44
               excess_ro_to_est = s80cap
               eqs80 = eqs80 - eflc44
c new s80 flow
               eqs80 = s80cap0
c or eqs80 = eqs80 + s80cap
c basin storage is excess runoff minus remaining capacity of s80
               bsnsto = eflc44 - s80cap
               excess_ro_to_sto = bsnsto
c remaining capacity of s80 is zero
               s80cap = 0.0
            ENDIF
         ELSE
c LOK stage is low enough, backflow to LOK possible
c might be limited by operational constraints,
c     may not be possible when lake stage is higher
c     than tailwater of S308
c assumption: maximum backflow is equal to design Q
            excess_ro_to_lok = eflc44
            eqs308 = eqs308 - eflc44
            s308cap = s308cap - eflc44
            eeflc44 = 0.0
            IF (s308cap .LT. 0.) THEN
c capacity of s308 exceeded, portion of excess runoff goes to tide
c put back flows
               s308cap = s308cap + eflc44
               excess_ro_to_lok = s308cap
               eqs308 = eqs308 + eflc44
c new s308 flow
               eqs308 = -1.0 * s308cap0
c remainder of excess runoff can go to tide
               eeflc44 = eflc44 - s308cap
               excess_ro_to_est = eeflc44
c remaining capacity of s308 is zero
               s308cap = 0.0
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c release rest of flow to tide, IF permissible
            eqs80 = eqs80 + eeflc44
            s80cap = s80cap - eeflc44
```

```
            IF (s80cap .LT. 0.0) THEN
               s80cap = s80cap + eeflc44
               excess_ro_to_est = s80cap
               eqs80 = eqs80 - eeflc44
               eqs80 = s80cap0
c or eqs80 = eqs80 + s80cap
               bsnsto = eeflc44 - s80cap
               excess_ro_to_sto = bsnsto
               s80cap = 0.0
            ENDIF
         ENDIF
      ENDIF
c
c C-44 BASIN demand accounting
c
c calculate c44 basin allocation
c
      alc44 = cbc44 * c44dmd
      IF (alc44 .EQ. 0.0) goto 20
c entire allocation is to be met by Lake Okeechobee
      eallok2 = alc44
c update estimated structure flow
      dmd_met_by_lok = eallok2
      eqs308 = eqs308 + eallok2
      IF (lok_split_option .EQ. 1) THEN
         IF (stglok .LE. stgref .AND. stagelo_res .GE. 2.0) THEN
            c44dmd_met_by_lokres = eallok2
         ELSE
            c44dmd_met_by_lokres = 0.0
         ENDIF
      ELSE
         c44dmd_met_by_lokres = 0.0
      ENDIF
      s308cap = s308cap - eallok2
      IF (s308cap.LT.0.0) THEN
         s308cap = s308cap + eallok2
         dmd_met_by_lok = s308cap
         eqs308 = eqs308 - eallok2
c
         eqs308 = s308cap0
c or eqs308 = eqs308 + s308cap
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         unmetdmd = eallok2 - s308cap
         c44dmd_met_by_lokres = c44dmd_met_by_lokres - unmetdmd
         c44dmd_met_by_lokres = amax1(c44dmd_met_by_lokres,0.0)
         s308cap = 0.0
      ENDIF
c
c WRITE summary for today
   20 WRITE(74,30)iyr,imo,ida,c44dmd,c44dmd,alc44,dmd_met_by_lok,
     $unmetdmd,c44ro0,c44ro0,excess_ro_to_est,excess_ro_to_lok,
     $excess_ro_to_sto,alsle0,est_dmd_met_by_local_ro, ealc44ro,eallok1,
     $sltinf,excess_local_ro,eqs308,eqs80
clb      ENDIF
      RETURN
   30 FORMAT(3i5,22f12.3)
      END
```

```
      SUBROUTINE stlucie2(iyr,imo,ida,ndays,c44dmd,c44ro,sltinf,sledmd,
     $cbc44,stglok,lok_supp_day,bflo_cap,bflo_frac, lok_supp_c44est,
     $c44dmd_met_by_lok,bsn2est_af, stgref,rmulti_seas_pred,
     $clim_threshold_est, cbslest,dmnd2_orig,ro2_orig)

C-----------------------------------------------------------------
      include 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'cstlucie.inc'
      INCLUDE 'lakwca.inc'
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      include 'opersched.inc'
      DIMENSION rmulti_seas_pred(12),clim_threshold_est(2)
c-----------------------------------------------------------------
C    This subroutine determines inflows/outflows between Lake Okeechobee,
C    St. Lucie basin,and St. Lucie estuary. This requires daily time
C    series of basin demand/runoff and a monthly time series of estuarine
C    demands.
c
c definition of terms: (all units in ac-ft/day)
c input:
c       c44dmd      = c44 basin demand
c       c44ro       = c44 basin runoff
c       sledmd      = caloosahatchee estuarine demand
c       cbc44       = fraction of c44 basin demand to be met
c                     as calculated in the supply-side management
c                     SUBROUTINE
c       stglok      = lake okeechobee stage
c
c output:
c       eallok1     = discharge from LOK used to meet estuarine demand
c       rmin_lok_stg_del_est = minimum LOK stage allowed to meet
c                              estuarine demand
c       alc44       = c44 basin allocation;
c                     c44 basin demand multiplied by the cutback cbc44
c       eflc44      = c44 basin runoff which can go to LOK, the estuarine
c                     or both
c       eqs80       = total flow thorugh s80 (excludes regulatory flows)
c       eqs308      = total flow thorugh s308 (excludes regulatory flows)
c
      REAL lok_supp0_day,lok_supp_day,bflo_cap,bflo_frac
      REAL lok_supp_c44est
c initialize allocations and estimated deliveries
      eallok1           = 0.0
      sledmd_met_by_lok = 0.0
      alc44             = 0.0
      eflc44            = 0.0
      c44dmd_met_by_lok = 0.0
      c44dmd_met_by_lokres = 0.0
      ro_to_lok         = 0.0
      ro_to_estuary     = 0.0
c     eqs80             = 0.0
c     eqs308            = 0.0
      sledmd0           = sledmd
      c44ro0            = c44ro
      IF (ida .EQ. 1) THEN
```

```
         lok_supp0_day = lok_supp_c44est/ndays
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
      CALL bpts(IDA,IMO,stglok,1,mzone,0,stgref_est)
      CALL bpts(ida,imo,10.0,1,n_fc_zones + 6, 1,
     $rmin_lok_stg_del_est_normf)
c
c adjust st lucie estuary demand by st lucie tributary inflow
c note: allocation will be based on sledmd adjusted for inflows
c       from other tributaries to estuary
      IF (sltinf .GT. sledmd)THEN
         est_dmd_met_by_local_ro = sledmd
         excess_local_ro = sltinf - sledmd
      ELSE
         est_dmd_met_by_local_ro = sltinf
         excess_local_ro = 0.0
      ENDIF
      sledmd = amax1(0.0,sledmd-sltinf)
c note: sledmd is the estuarine demand to be met by Lake Okeechobee
c       and c44 basin.  It can be cutback by using the cutback fraction
c       from ssm or by other means but will not be subject to any cutback
c       in this version because the target supplemental deliveries are
c       preprocessed based on known runoff/demand conditions in St. Lucie
c       basin and tributaries
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (lok_supp_day .GT. 0.) THEN
c supplementary release from LOK required
         IF ((stglok .GT. rmin_lok_stg_del_est .AND. iclimate_opt .EQ.
     $   0) .OR. (stglok .GT. rmin_lok_stg_del_est .AND.
     $   ((rmulti_seas_pred(imo) .GT. clim_threshold_est(1) .AND.
     $   stglok .GE. rmin_lok_stg_del_est_normf .AND. mzone .GE.
     $   mzone_puls_top).OR. (rmulti_seas_pred(imo) .GT.
     $   clim_threshold_est(2) .AND. stglok .LT.
     $   rmin_lok_stg_del_est_normf) .OR. (mzone .LT. mzone_puls_top
     $   .AND. no_puls_zones .GT. 0) .OR. (mzone .LE. n_fc_zones .AND.
     $   no_puls_zones .EQ. 0)))) THEN

            eallok1 = lok_supp_day*cbslest
            eallokres2 = 0.0
         ELSE
            eallok1 = 0.0
            IF (lok_split_option .EQ. 1 .AND. stagelo_res .GE. 2.0) THEN
               eallokres2 = lok_supp_day
            ELSE
               eallokres2 = 0.0
            ENDIF
         ENDIF
c note: eallok1 can be made subject to cutback ==> DO this in future versions
c update estimated structure flows
         eqs80 = eqs80 + eallok1 + eallokres2
         s80cap = s80cap - eallok1 - eallokres2
         eqs308 = eqs308 + eallok1 + eallokres2
         s308cap = s308cap - eallok1 - eallokres2
         sledmd_met_by_lok = eallok1 + eallokres2
      ELSE
```

```
         eallokres2 = 0.0
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c C-44 basin runoff, IF any, can backflow, IF LOK stage is low enough, or go to
c estuary as excess basin runoff (note: This runoff may be considered as excess
c or partial excess relative to today's estuarine demand but may all be needed
c to satisfy the estuarine's total monthly demand.)
      IF (c44ro .GT. 0.) THEN
         eflc44 = c44ro
         IF (lok_split_option .EQ. 1) THEN
            stglokup = stagelo_res
            stg_ref  = 14.5
         ELSE
            stglokup = stglok
            CALL bpts(ida,imo,10.0,1,n_fc_zones,1,stg_ref)
         ENDIF
         CALL bpts(ida,imo,10.0,1,n_fc_zones + 8,1,stg_backflw_stl)
         stg_backflow = stg_backflw_stl
         IF (stglokup.GT.stg_backflow) THEN
c LOK stage too high, release to tide
            ro_to_estuary = eflc44
            eqs80 = eqs80 + eflc44
            s80cap = s80cap - eflc44
         ELSE
c LOK stage is low enough, backflow to LOK possible
            IF (bflo_cap .GE. bflo_frac * eflc44) THEN
c release all runoff to LOK within allowable fraction
               ro_to_lok = bflo_frac*eflc44
               eqs308 = eqs308 - ro_to_lok
               s308cap = s308cap - ro_to_lok
c               s308cap = s308cap + ro_to_lok
               bflo_cap = bflo_cap - ro_to_lok
c release remaining runoff to estuary, assume no c44 basin runoff stays in stora
               ro_to_estuary = eflc44 - ro_to_lok
               eqs80 = eqs80 + ro_to_estuary
               s80cap = s80cap - ro_to_estuary
            ELSE
c release portion of runoff to LOK up to bflo_cap within allowable fraction
               ro_to_lok = amin1(bflo_frac * eflc44,bflo_cap)
               eqs308 = eqs308 - ro_to_lok
               s308cap = s308cap - ro_to_lok
c               s308cap = s308cap + ro_to_lok
               IF (ro_to_lok .EQ. bflo_cap) THEN
c backflow capacity is reduced to zero
                  bflo_cap = 0.0
               ELSE
                  bflo_cap = bflo_cap - ro_to_lok
               ENDIF
c release remaining runoff to estuary
               ro_to_estuary = eflc44 - ro_to_lok
               eqs80 = eqs80 + ro_to_estuary
               s80cap = s80cap + ro_to_estuary
            ENDIF
         ENDIF
      ENDIF
c
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (c44dmd .GT. 0.) THEN
         alc44 = cbc44*c44dmd
c entire allocation to be met by Lake Okeechobee
         c44dmd_met_by_lok = alc44
c update structure flow
         IF (lok_split_option .EQ. 1) THEN
            IF (stglok .LE. stgref .AND. stagelo_res .GE. 2.0) THEN
               c44dmd_met_by_lokres = alc44
            ELSE
               c44dmd_met_by_lokres = 0.0
            ENDIF
         ELSE
            c44dmd_met_by_lokres = 0.0
         ENDIF
         eqs308 = eqs308 + c44dmd_met_by_lok
         s308cap = s308cap - c44dmd_met_by_lok
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c WRITE summary for today
   10 WRITE(74,20)iyr,imo,ida,dmnd2_orig,c44dmd,alc44,c44dmd_met_by_lok,
     $ro2_orig,c44ro0,ro_to_lok,ro_to_estuary,stglok,sledmd0,
     $excess_local_ro,sltinf,lok_supp0_day,lok_supp_day, bflo_cap,
     $eqs308,eqs80,bsn2est_af
clb      ENDIF
      RETURN
   20 FORMAT(3i5,22f12.3)
      END
```

small_res_gw_flow.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: small_res_gw_flow.F,v 1.8 2005/06/10 13:29:19 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/small_res_gw_flow.F,v $
C   $Name: rel-5-5 $
C
C
C   PURPOSE:
C    SOLVE THE GROUNDWATER EQUATION WITHIN SMALL RESERVOIRS PASSING
C    THROUGH MORE THAN ONE GRID CELL AND TREATED AS SEPARATE ENTITIES.
C
C   VARIABLE DEFINITIONS
C    AQDEP_in_res     THICKNESS OF AQUIFER BELOW MSL IN RESERVOIR (FT)
C    H         GROUNDWATER STAGE AT EACH NODE
C    IXM1      SUBSCRIPT OF NODE IN X-1 DIRECTION
C    IXP1          "     "    "     X+1     "
C    IYM1          "     "    "     Y-1     "
C    IYP1          "     "    "     Y+1     "
C    MAXXT     MAXIMUM NUMBER OF NODES IN ANY ROW
C    NCA       (1,2) INDICATOR TO CONTROL ALT. DIR. SOLUTION
C    aqperm_in_res     PERMEABILITY OF AQUIFER  IN RESERVOIR
C    TX ,TY    HARMONIC MEANS OF TRANSMISSIVITY
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE small_res_gw_flow (iresnum, ells_in_res, stage_res,
     & pond_in_res, prev_stage_res, aqperm_in_res, aqdep_in_res, nca_r)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'budg.inc'
```

```
      DIMENSION aqdep_in_res(max_ncells,max_n_res_in_cell),
     & aqperm_in_res(max_ncells,max_n_res_in_cell),
     & ells_in_res(max_ncells,max_n_res_in_cell),
     & prev_stage_res(max_ncells,max_n_res_in_cell),
     & pond_in_res(max_ncells,max_n_res_in_cell),
     & stage_res(max_ncells,max_n_res_in_cell)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

       NCA_R = NCA_R + 1
       IF (NCA_R .GT. 2) NCA_R = 1
       IF (NCA_R .EQ. 1) THEN
         JJ = 1
         KK = NNODES(iresnum)-1
         INCR = 1
       ELSE
         JJ = NNODES(iresnum)
         KK = 2
         INCR = -1
       ENDIF
       DO IN = JJ,KK,INCR
c  reservoir cell loop
c
          IF (NCA_R .EQ. 1 ) THEN
            INEXT = IN + 1
          ELSE
            INEXT = IN - 1
          ENDIF
          IF (idirect(iresnum,in) .eq. 1) THEN
            RLGTH_FLW = DX
             IF (node_reserv(iresnum,inext) .gt.
     $       node_reserv(iresnum,in)) THEN
               signr = 1.0
               noder = node_reserv(iresnum,in)
             ELSE
               signr = -1.0
               noder = node_reserv(iresnum,inext)
             ENDIF
          ELSE
            RLGTH_FLW = DY
             IF (node_reserv(iresnum,inext) .lt.
     $       node_reserv(iresnum,in)) THEN
               signr = 1.0
               noder = node_reserv(iresnum,in)
             ELSE
               signr = -1.0
               noder = node_reserv(iresnum,inext)
             ENDIF
          ENDIF
          WDTHGW = width_of_res(iresnum)*5280.
C
          DO ir1 = 1,noresincell(node_reserv(iresnum,in))
            IF (iresnum .eq. ires_index_in_cell
     $      (node_reserv(iresnum,in),ir1)) THEN
              iru = ir1
              GO TO 10
            ENDIF
```

```
          ENDDO
 10       DO ir2 = 1,noresincell(node_reserv(iresnum,inext))
            IF (iresnum .eq. ires_index_in_cell
     $      (node_reserv(iresnum,inext),ir2)) THEN
              ird = ir2
              GO TO 20
            ENDIF
          ENDDO
C--5----0----5----0----5----0----5----0----5----0----5----0----5----0--
C-----------------
C  FIRST FIND CURRENT TRANSMISSIVITY VALUES FOR THESE GRID POINTS
C  ALONG RESERVOIR
C-----------------
 20       NODE = node_reserv(iresnum,in)
          NODED = node_reserv(iresnum,inext)
          TH = aqperm_in_res(NODE,iru)*(AQDEP_in_res(NODE,iru)
     $    +stage_res(node,iru))
          YP = aqperm_in_res(NODED,ird)*(AQDEP_in_res(NODED,ird)
     $    + stage_res(noded,ird))
          TYPAV = (TH + YP) / 2.0
          HAVG = (stage_res(NODE,iru)+prev_stage_res(NODE,iru)) / 2.0
          HDYP1 = HAVG - stage_res(NODED,ird)
C-----------------
C  NOW CALCULATE VOLUME OF GW FLOW ACROSS CELL BOUNDARY FACE
C-----------------
          HMVOL(NODER,idirect(iresnum,in))=HMVOL(NODER,idirect(iresnum
     $    ,in)) + signr*HDYP1*TYPAV*width_of_res(iresnum)*5280.* DT
     $    / RLGTH_FLW
          HMVOL_dailyr = HDYP1 * TYPAV * width_of_res(iresnum)
     $    *5280. * DT / RLGTH_FLW
c
c  adjust upstream and downstream reserv stages
c
          depth_to_wtu = amax1(ells_in_res(node,iru)
     $    - stage_res(node,iru),0.0) * s(node)
          depth_to_wtd = amax1(ells_in_res(noded,ird)
     $    - stage_res(noded,ird),0.0) * s(noded)
          dpth_of_gwflo = HMVOL_dailyr/(WDTHGW * RLGTH_FLW)
c         print *,dpth_of_gwflo,HMVOL_dailyr,WDTHGW,HAVG
c    $       ,RLGTH_FLW,stage_res(node,iru),stage_res(noded,ird)
c    $       ,'   GWGWGWGWGW'
          IF (HMVOL_dailyr .gt. 0.00001) THEN
             stage_res(node,iru) = stage_res(node,iru)
     $       - amin1(dpth_of_gwflo,pond_in_res(node,iru))
     $       - amax1(dpth_of_gwflo-pond_in_res(node,iru),0.0)
     $       / s(node)
             pond_in_res(node,iru) = pond_in_res(node,iru)
     $       - amin1(dpth_of_gwflo,pond_in_res(node,iru))
             stage_res(noded,ird) = stage_res(noded,ird)
     $       + amin1(dpth_of_gwflo,depth_to_wtd)/s(noded)
     $       + amax1(dpth_of_gwflo-depth_to_wtd,0.0)
             pond_in_res(noded,ird) = amax1(stage_res(noded,ird)
     $       - ells_in_res(noded,ird),0.0)
          ELSE
             sgn = -1.0
             stage_res(node,iru) = stage_res(node,iru)
     $       + amin1(sgn*dpth_of_gwflo,depth_to_wtu)/s(node)
```

```
     $       + amax1(sgn*dpth_of_gwflo-depth_to_wtu,0.0)
             pond_in_res(node,iru) = amax1(stage_res(node,iru)
     $       - ells_in_res(node,iru),0.0)
             stage_res(noded,ird) = stage_res(noded,ird)
     $       - amin1(sgn*dpth_of_gwflo,pond_in_res(noded,ird))
     $       - amax1(sgn*dpth_of_gwflo-pond_in_res(noded,ird),0.0)
     $       / s(noded)
             pond_in_res(noded,ird) = pond_in_res(noded,ird)
     $       - amin1(pond_in_res(noded,ird),sgn*dpth_of_gwflo)
         ENDIF
c
c  end of reservoir loop
       ENDDO
       RETURN
       END
```

small_res_stor.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: small_res_stor.F,v 1.11 2005/06/10 13:29:19 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/small_res_stor.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE COMPUTES AVAILABLE STORAGE FOR RESERVOIRS TREATED
C   AS SEPARATE ENTITIES WITHIN MODEL DOMAIN.
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE small_res_stor (ia, volwtr, qoutmx, sumrain,
     & total_afact, prev_stage_res, deltastor, tvolwtrl, avvolgr_mx,
     & iseas)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'asr2.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'reuse.inc'

      DIMENSION avvolgr_mx(max_n_res,max_nostr_res),
     & deltastor(max_n_res),
     & prev_stage_res(max_ncells,max_n_res_in_cell),
     & qoutmx(max_n_res,max_nostr_res), sumrain(max_ncells),
     & total_afact(max_ncells), tvolwtrl(max_n_res), volwtr(max_n_res)
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  DETERMINE THE MAXIMUM AVAILABLE STORAGE FOR EACH SMALL RESERVOIR
C  TREATED AS SEPARATE ENTITY IN MODEL GRID DOMAIN

      volwtr(ia)=0.0
      deltastor(ia)=0.0
      DO ij=1,nasr_from_res(ia)
        avail_vol_to_asr(ia,ij)=0.0
      END DO
      mean_ponding_depth(ia)=0.0
      DO 10 iy=maxy,1,-1
        DO 10 ix=minx(iy),maxx(iy)
        node=ix-minx(iy)+1+isum(iy)
        DO ir = 1,noresincell(node)
        IF (ia .eq. ires_index_in_cell(node,ir)) THEN
           mean_ponding_depth(ia)=mean_ponding_depth(ia)
     &     +pond_in_res(node,ir)/nnodes(ia)
           IF (stage_res(node,ir).GT.ells_in_res(node,ir)) THEN
              volwtr(ia)=volwtr(ia)+(stgmx(ia)-stage_res(node
     &        ,ir))*sfactor(ia)*gdar/86400.
           ELSE
              volwtr(ia)=volwtr(ia)+amax1(stgmx(ia)-ells_in_res(node
     &        ,ir),0.0)*sfactor(ia)*gdar/86400.+(amin1(stgmx(ia)
     &        ,ells_in_res(node,ir))-stage_res(node,ir))*s(node)*gdar
     &        *sfactor(ia)/86400.
              pond_in_res(node,ir) = 0.0
           ENDIF
           deltastor(ia) = deltastor(ia) + (amax1(stage_res(node,ir)
     &     ,ells_in_res(node,ir)) - amax1(prev_stage_res(node,ir)
     &     ,ells_in_res(node,ir)) + (amin1(stage_res(node,ir)
     &     ,ells_in_res(node,ir)) - amin1(prev_stage_res(node,ir)
     &     ,ells_in_res(node,ir)))*s(node)) * sfactor(ia)*gdar/43560.
c          if (resname(ia).eq.'C4IMP ') print *,deltastor(ia)
c    &     ,stage_res(node,ir),prev_stage_res(node,ir),'  DELTASTR '
c    &     ,ells_in_res(node,ir),sfactor(ia),volwtr(ia)
        ENDIF
        ENDDO
  10  CONTINUE
      DO i=1,noutstr(ia)
         IF (res_out_type(ia,i).ne.'WSPLY') THEN
            DO ip=1,npumptores
               IF (iwmads(ia,i).eq.ibsnres(ip)) THEN
                  qoutmx(ia,i)=pumpcap_res(ip)
               END IF
            END DO
            IF (wrexp(ia,i,2).eq.1.0) THEN
               IF (inodopt(ia,i,3).eq.1) THEN
                  IF (iwmads(ia,i).le. ntotres) THEN
                     DO ir = 1,noresincell(nodepw(ia,i,1))
                     IF (iwmads(ia,i).eq. ires_index_in_cell
     &               (nodepw(ia,i,1),ir))  THEN
                     IF (sfactor(iwmads(ia,i)) .lt. sfactmin .and.
     &               ires_small_sim(iwmads(ia,i)) .eq. 'YES') then
                        tail = stage_res(nodepw(ia,i,1),ir)
                     ELSE
                       IF (h(nodepw(ia,i,1))+pond(nodepw(ia,i,1))
```

```
     &                  .le.ells(nodepw(ia,i,1))) THEN
                           afact=1.0
                        ELSE
                           afact=total_afact(iwmads(ia,i))
                        ENDIF
                        adjust_for_ovlf=sum_chg_dpth_ovl_flw(nodepw(ia,
     &                  i,1))
                        term1_for_adjust_area=(sumet_outside_res(nodepw
     &                  (ia,i,1))+seep_loss_res(nodepw(ia,i,1))
     &                  -sumrain(nodepw(ia,i,1))/12.0
     &                  -sum_chg_depth_levee_spg(nodepw(ia,i,1)))*
     &                  ((1.0-afact)/afact)
                        term2_for_adjust_area=
     &                  (outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &                  outf_seep_loc_to_res_cell(nodepw(ia,i,1))+
     &                  adjust_for_ovlf)/afact
                        deptgw=(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))*
     &                  s(nodepw(ia,i,1))
                        IF (pond(nodepw(ia,i,1)).gt.deptgw) THEN
                           tail=pond(nodepw(ia,i,1))-deptgw+ells
     &                     (nodepw(ia,i,1))
                        ELSE
                           tail=(pond(nodepw(ia,i,1))-deptgw)/
     &                     s(nodepw(ia,i,1))+ells(nodepw(ia,i,1))
                        END IF
                        IF ((tail.ge.ells(nodepw(ia,i,1))
     &                  .or.(stage_res(nodepw(ia,i,1),ir).ge.ells(nodepw
     &                  (ia,i,1)).and.iwmads(ia,i).le.ntotres)).and.
     &                  nodepw(ia,i,1).ne.idnstrm(ia,i)) THEN
                           tail=ells(nodepw(ia,i,1))+pond(nodepw(ia,i
     &                     ,1))/afact+term1_for_adjust_area-
     &                     adjust_dur_excess(nodepw(ia,i,1))-
     &                     (outf_fc_frm_res_to_cell(nodepw(ia,i,1))+
     &                     outf_seep_loc_to_res_cell(nodepw(ia,i,1)))
     &                     /afact-sum_chg_dpth_ovl_flw(nodepw(ia,i,1))
     &                     /afact+daily_chg_dpth_ovl_vol1(nodepw(ia,i,1))
     &                     /afact
!
                           tail=tail-adjust_for_area_m1(nodepw(ia
     &                     ,i,1))
                        END IF
                        tail=amax1(tail,ells(nodepw(ia,i,1)))
                     ENDIF
                     ENDIF
                     ENDDO
                  ELSE
                     deptgw=(ells(nodepw(ia,i,1))-h(nodepw(ia,i,1)))
     &               * s(nodepw(ia,i,1))
                     IF (pond(nodepw(ia,i,1)).gt.deptgw) THEN
                        tail=pond(nodepw(ia,i,1))-deptgw+ells(nodepw
     &                  (ia,i,1))
                     ELSE
                        tail=(pond(nodepw(ia,i,1))-deptgw)/s(nodepw
     &                  (ia,i,1))+ells(nodepw(ia,i,1))
                     END IF
                     IF (nodepw(ia,i,1) .ne. idnstrm(ia,i)) then
                       tail = amax1(tail,ells(nodepw(ia,i,1)))
```

```
                  ENDIF
               ENDIF
            ELSE
               IF (ircnl(ia,i,1).le.nch) THEN
                 IF (nodcr(ircnl(ia,i,1)) .gt. 1) then
                  diff_canlr_stg_frm_ds_end = float(nodcr(ircnl(ia,i
     &            ,1))-NODEPOS_CANLR(ia,i,1))/float(nodcr(ircnl(ia,i
     &            ,1))-1)*hdc(ircnl(ia,i,1),iseas)
                 ELSE
                  diff_canlr_stg_frm_ds_end = 0.0
                 ENDIF
                  tail=chdep(ircnl(ia,i,1))+diff_canlr_stg_frm_ds_end
               ELSE
                  tail=crstelev(ia,i,2)
               END IF
            ENDIF
            qoutmx(ia,i)=weircff(ia,i,2)*(stgmx(ia)-tail)
c           if (resname(ia).eq. 'C4IMP ') print *,qoutmx(ia,i),tail
c    &         ,stgmx(ia),' CCCC'
         ELSE
            IF (type(ia,i).eq.'GRAVITY') THEN
              wcrsth=amax1(crstelev(ia,i,2),tail)
            ELSE
              wcrsth=crstelev(ia,i,2)
            END IF
            IF (wcrsth.gt.crstelev(ia,i,2)) THEN
              qfactor=0.90
            ELSE
              qfactor=1.0
            END IF
            IF (stgmx(ia).gt.crstelev(ia,i,2)) THEN
              qoutmx(ia,i)=weircff(ia,i,2)*(stgmx(ia)-crstelev(ia,i,
     &         2))**wrexp(ia,i,2)*qfactor
            ELSE
              qoutmx(ia,i)=0.0
            END IF
            qoutmx(ia,i) = amin1(qoutmx(ia,i),avvolgr_mx(ia,i))
            IF (iwmads(ia,i).lt. ntotres) then
               qoutmx(ia,i) = amin1(qoutmx(ia,i)
     &         ,tvolwtrl(iwmads(ia,i)))
               tvolwtrl(iwmads(ia,i)) = tvolwtrl(iwmads(ia,i))
     &         - qoutmx(ia,i)
            ENDIF
         ENDIF
         IF (inodopt(ia,i,3).ne.1.and.ircnl(ia,i,1).le.nch) THEN
           IF (itrgcnl(ia,i,1) .le. nch) THEN
             if (nodcr(itrgcnl(ia,i,1)) .gt. 1) then
               diff_stg_cnl_trg = float((nodcr(itrgcnl(ia,i,1))-
     &         nodepos_rcnl_trg(ia,i,1)))/float(nodcr(itrgcnl
     &         (ia,i,1))-1)*hdc(itrgcnl(ia,i,1),iseas)
             else
               diff_stg_cnl_trg = 0.0
             endif
             dn_cnl_stg_trg = chdep(itrgcnl(ia,i,1))+diff_stg_cnl_trg
           ELSE
             if (icell_trg_loc(ia,i,1) .gt. 0) THEN
               dn_cnl_stg_trg = h(icell_trg_loc(ia,i,1))
```

```
     &            + pond(icell_trg_loc(ia,i,1))
                else
                  dn_cnl_stg_trg = -901.0
                endif
              ENDIF
              IF (dn_cnl_stg_trg.lt.dwnstgmx(ia,i,1)+
     &        offset_reg_res(ia,i,1).and.dn_cnl_stg_trg
     &        .ge.dwnstgmx(ia,i,1)) THEN
                qoutmx(ia,i)=1.0/offset_reg_res(ia,i,1)*(dwnstgmx(ia,i
     &          ,1)+offset_reg_res(ia,i,1)-dn_cnl_stg_trg)*qoutmx(ia,
     &          i)
              ELSE IF (dn_cnl_stg_trg.ge.dwnstgmx(ia,i,1)+
     &          offset_reg_res(ia,i,1)) THEN
                qoutmx(ia,i)=0.0
              END IF
c             if (resname(ia).eq.'C4IMP') print *,dn_cnl_stg_trg
c    &        ,offset_reg_res(ia,i,1),dwnstgmx(ia,i,1)
c    &        ,qoutmx(ia,i),'  C4IMPSTOR  ',volwtr(ia)
c    &        ,cnm(itrgcnl(ia,i,1)),diff_stg_cnl_trg
c    &        ,hdc(itrgcnl(ia,i,1),iseas),chdep(itrgcnl(ia,i,1))
c    &        ,nodepos_rcnl_trg(ia,i,1),nodcr(itrgcnl(ia,i,1))
            END IF
          ELSE
            qoutmx(ia,i)=0.0
          END IF
          volwtr(ia)=volwtr(ia)+qoutmx(ia,i)
        END DO
        rmax_outflow_asr=0.0
        IF (nasr_from_res(ia).gt.0) THEN
          DO j=1,nasr_from_res(ia)
            rmax_outflow_asr=rmax_outflow_asr+asr_incap(asrnum(2,ia,j,1)
     &       )*1.532
          END DO
        END IF
        volwtr(ia)=volwtr(ia)+rmax_outflow_asr
c       if (resname(ia) .eq.'C9RES') print *,qoutmx(ia,i)
c    &  ,volwtr(ia),rmax_outflow_asr,'  IN SMALL RES STOR'
      RETURN
      END
```

small_reserv_ovlnf.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: small_reserv_ovlnf.F,v 1.6 2003/07/03 20:07:46 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/small_reserv_ovlnf.F,v $
C   $Name: rel-5-5 $
C
C
C  SUBROUTINE CALCULATES OVERLAND FLOW WITHIN SMALLER RESERVOIRS
C  TREATED INDEPENDENTLY
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE small_reserv_ovlnf (iresnum, ells_in_res, pond_in_res,
     & stage_res)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'abc.inc'

      DIMENSION ells_in_res(max_ncells,max_n_res_in_cell),
     & pond_in_res(max_ncells,max_n_res_in_cell),
     & stage_res(max_ncells,max_n_res_in_cell)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      deten_in_res = 0.1
      DTS = DT*86400. / NTSTEP_OV
      nodell = 24 - minx(14) + 1 + isum(14)

C  time loop

      DO ITIME = 1,NTSTEP_OV
```

```
        IF (ITIME .EQ. 1 .OR. ITIME .EQ. 3) THEN
           IT1 = 1
           IT2 = nnodes(iresnum) - 1
           INCR = 1
        ELSE
           IT1 = nnodes(iresnum)
           IT2 = 2
           INCR = -1
        ENDIF
c
        DO IN = IT1,IT2,INCR
c  reservoir cell loop
           IF (INCR .GT. 0) THEN
             INEXT = IN + 1
           ELSE
             INEXT = IN - 1
           ENDIF
           IF (idirect(iresnum,in) .eq. 1) THEN
             RLGTH_FLW = DX
             IF (node_reserv(iresnum,inext) .gt.
     $       node_reserv(iresnum,in)) THEN
               signr = 1.0
               noder = node_reserv(iresnum,in)
             ELSE
               signr = -1.0
               noder = node_reserv(iresnum,inext)
             ENDIF
           ELSE
             RLGTH_FLW = DY
             IF (node_reserv(iresnum,inext) .lt.
     $       node_reserv(iresnum,in)) THEN
               signr = 1.0
               noder = node_reserv(iresnum,in)
             ELSE
               signr = -1.0
               noder = node_reserv(iresnum,inext)
             ENDIF
           ENDIF
           WDTHOV = width_of_res(iresnum)*5280.
C
           DO ir1 = 1,noresincell(node_reserv(iresnum,in))
             IF (iresnum .eq. ires_index_in_cell
     $       (node_reserv(iresnum,in),ir1)) THEN
               iru = ir1
               GO TO 10
             ENDIF
           ENDDO
 10        DO ir2 = 1,noresincell(node_reserv(iresnum,inext))
             IF (iresnum .eq. ires_index_in_cell
     $       (node_reserv(iresnum,inext),ir2)) THEN
               ird = ir2
               GO TO 20
             ENDIF
           ENDDO
 20        HU = ells_in_res(node_reserv(iresnum,in),iru) +
     $     pond_in_res(node_reserv(iresnum,in),iru)
           HD = ells_in_res(node_reserv(iresnum,inext),ird) +
```

```
     $     pond_in_res(node_reserv(iresnum,inext),ird)
           HM = (pond_in_res(node_reserv(iresnum,in),iru)
     $     + pond_in_res(node_reserv(iresnum,inext),ird)) /2.0
c          print *,HU,HD,HM,iru,ird,WDTHOV,IN,INEXT,'  OVOVOVOV  '
c    $     ,node_reserv(iresnum,in),node_reserv(iresnum,inext)
c
c  ovlnf occurs if meam depth of flow is greater than detention depth
c
           IF (HM .GT. deten_in_res) THEN
             ZNU = OFML(lutyp_res_loc(iresnum,in),1)
     $       * HM ** OFML(lutyp_res_loc(iresnum,in),2)
             ZND = OFML(lutyp_res_loc(iresnum,inext),1)
     $       * HM ** OFML(lutyp_res_loc(iresnum,inext),2)
             ZN = (ZNU + ZND)/2.0
             ZN = amax1(ZN,rmin_ofml(lutyp_res_loc(iresnum,in)))
             OFM = 1.49 / ZN
c            print *,ZNU,ZND,ZN,OFM,HU,HD,lutyp_res_loc(iresnum,in)
c    $       ,'       VOVOVOVOVO  ',RLGTH_FLW
             IF (HU .GT. HD ) THEN
c  upstream head greater than downstream head
               VOF = OFM * (HU - HD)/SQRT(RLGTH_FLW)*HM ** (2./3.)
               VOLOV = VOF * DTS * WDTHOV * HM
               avg_stg = (HU + HD) / 2.0
               OVMAX = (avg_stg - HD) * WDTHOV * RLGTH_FLW
               OVTEST = (pond_in_res(node_reserv(iresnum,in),iru)
     $         - deten_in_res) * WDTHOV * RLGTH_FLW
               OVTEST = AMAX1(OVTEST,0.0)
               IF (OVMAX .GT. OVTEST) OVMAX = OVTEST
c              print *,OVTEST,OVMAX,avg_stg,HD,'  POPOPOPOPO ',VOLOV
c    $         ,VOF
               pond_in_res(node_reserv(iresnum,in),iru) =
     $         pond_in_res(node_reserv(iresnum,in),iru) -
     $         amin1(volov,ovmax)/(WDTHOV * RLGTH_FLW)

               stage_res(node_reserv(iresnum,in),iru) = stage_res(
     $         node_reserv(iresnum,in),iru) - amin1(volov,ovmax)
     $         /(WDTHOV * RLGTH_FLW)
c
               depth_to_wt = (ells_in_res(node_reserv(iresnum,inext)
     $         ,ird) - stage_res(node_reserv(iresnum,inext),ird)) *
     $         s(node_reserv(iresnum,inext))
               depth_to_wt = amax1(depth_to_wt,0.0)
               dpth_of_ovflo = amin1(volov,ovmax)/(WDTHOV * RLGTH_FLW)
               stage_res(node_reserv(iresnum,inext),ird) =
     $         stage_res(node_reserv(iresnum,inext),ird) +
     $         amin1(dpth_of_ovflo,depth_to_wt)/s(node_reserv(iresnum
     $         ,inext)) + amax1(dpth_of_ovflo - depth_to_wt,0.0)
               pond_in_res(node_reserv(iresnum,inext),ird) =
     $         amax1(stage_res(node_reserv(iresnum,inext),ird) -
     $         ells_in_res(node_reserv(iresnum,inext),ird),0.0)
               OVLFLO(noder,idirect(iresnum,in))
     $         = OVLFLO(noder,idirect(iresnum,in))
     $         + signr* amin1(volov,ovmax)
             ELSE
c  downstream head greater than upstream head
               VOF = OFM * (HD - HU)/SQRT(RLGTH_FLW)*HM ** (2./3.)
               VOLOV = VOF * DTS * WDTHOV * HM
```

```
          avg_stg = (HU + HD) / 2.0
          OVMAX = (HD - avg_stg) * WDTHOV * RLGTH_FLW
          OVTEST = (pond_in_res(node_reserv(iresnum,inext),ird)
$         - deten_in_res) * WDTHOV * RLGTH_FLW
          OVTEST = AMAX1(OVTEST,0.0)
          IF (OVMAX .GT. OVTEST) OVMAX = OVTEST
          pond_in_res(node_reserv(iresnum,inext),ird) =
$         pond_in_res(node_reserv(iresnum,inext),ird) -
$         amin1(VOLOV,OVMAX)/(WDTHOV * RLGTH_FLW)
          stage_res(node_reserv(iresnum,inext),ird) =
$         stage_res(node_reserv(iresnum,inext),ird) -
$         amin1(VOLOV,OVMAX)/(WDTHOV * RLGTH_FLW)
          depth_to_wt = (ells_in_res(node_reserv(iresnum,in)
$         ,iru) - stage_res(node_reserv(iresnum,in),iru)) *
$         s(node_reserv(iresnum,in))
          depth_to_wt = amax1(depth_to_wt,0.0)
          dpth_of_ovflo = amin1(volov,ovmax)/(WDTHOV * RLGTH_FLW)
          stage_res(node_reserv(iresnum,in),iru) =
$         stage_res(node_reserv(iresnum,in),iru) +
$         amin1(dpth_of_ovflo,depth_to_wt)/s(node_reserv(iresnum
$         ,in)) + amax1(dpth_of_ovflo - depth_to_wt,0.0)
          pond_in_res(node_reserv(iresnum,in),iru) =
$         amax1(stage_res(node_reserv(iresnum,in),iru) -
$         ells_in_res(node_reserv(iresnum,in),iru),0.0)
          OVLFLO(noder,idirect(iresnum,in))
$         = OVLFLO(noder,idirect(iresnum,in))
$         - signr * amin1(VOLOV,OVMAX)
        ENDIF
      ENDIF
   ENDDO                  ! end of reservoir cell loop
 ENDDO                    ! end of time loop
 RETURN
 END
```

smallres.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: smallres.inc,v 1.4 2003/07/03 20:08:05 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/smallres.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /smallres/ total_small_afact(max_ncells)
```

spec_canl_dep_struc_flw.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: spec_canl_dep_struc_flw.F,v 1.28 2005/06/10 13:29:20 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/spec_canl_dep_struc_flw.F,v $
C   $Name: rel-5-5 $
C
C   THIS SUBROUTINE COMPUTES DISCHARGES THROUGH CANAL OUTLET STRUCTURES
C   THAT REQUIRE UNIQUE CODE.
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE spec_canl_dep_struc_flw (n, ind1, idncl, iter, logdisg,
     & conv, rain, tivolwtr, capacws, capacfl, volwtr, prevqu,
     & tempf_volwtr, tempf_weircff, rloc_excess_for_ws, tdqu_dn,
     & rloc_excss_to_canl_next_step, canl_rf_avg, s333s334rg_offset,
     & rlist_lvname, convey_canal_names, struc_name_meas, s334_fc,
     & iroute_to_wca3a_frm_holy, iseep_lim_s356_opt, tempvol,
     & frac_avail_cap_s333, is334_open_flg, s356_to_l29, s356_to_cell,
     & s337fc, cout, finstor, finstor_res, jday, tinflow,
     & outflow_from_3a, tdmnd_ds, ws_to_l8_via_s352, wratio, imgt,
     & nconvey_total, pctmet, rmflws12, rmflws333, stagelo_beg,
     & outflow_from_3a_temp, s12_reg_total, s12_env_total,
     & flow_to_lbn_frm_c11, s333_reg_lec, s355prevout, prev_totf,
     & s333_reg_lec_abve_sch, s333_reg_lec_bel_sch, s333ftargprev,
     & total_flow_to_mcnl, ihly_out_count, stgref, total_flow_s12,
     & rfls155a, prev_total_flow_s12, hws, tws, tdmnd, crmin,
     & s336_to_biscbay, flow_to_acmres_frm_basin, lpmps309,
     & stgdiff_canal_tw, qinfldn, coutlim, avvol_dns334,
     & add_s333_fc_s334, stage_g3273,total_small_afact,
     & lflwth, l334rg, lwsdel,add_dmndagl8,prev_voldn_s334,
     &outf_s9a,out_flow_east,out_flow_east0,iroute_acme_fc_to_sta1e_opt,
     & frac_s319_sta1ew,frac_s319_sta1ee,iroute_l8rnf_to_sta_opt)

      INCLUDE  'wmm_par.inc'
      INCLUDE  'abc.inc'
```

```
      INCLUDE  'c1.inc'
      INCLUDE  'c2.inc'
      INCLUDE  'routc.inc'
      INCLUDE  'stat.inc'
      INCLUDE  'wrout.inc'
      INCLUDE  'wcaot.inc'
      INCLUDE  'stas.inc'
      INCLUDE  'agdata.inc'
      INCLUDE  'lakwca.inc'
      INCLUDE  'et.inc'
      INCLUDE  'environ.inc'
      INCLUDE  'resadj.inc'
      INCLUDE  'wcasim.inc'
      INCLUDE  'resws.inc'
      INCLUDE  'opersched.inc'
      INCLUDE  'lokres.inc'
      INCLUDE  'resbudg.inc'
      INCLUDE  'stormd.inc'
      INCLUDE  'dualops.inc'

      DIMENSION avvolws(max_n_wcas,max_n_conv_cnl_wca),
     & canl_rf_avg(max_ncnls), capacfl(max_n_seas,max_n_substr),
     & capacws(max_n_seas), crmin(max_ncnls,max_n_seas),
     & etout(max_ncnls), ilgth(max_ncnlloc), nodec(max_ncnlloc),
     & outflow_from_3a(max_n_conv_cnl_wca),
     & outflow_from_3a_temp(max_n_conv_cnl_wca), prevqu(max_ncnls),
     & rain(max_ncells), rfin(max_ncnls),
     & rloc_excess_for_ws(max_n_serv_area),
     & rloc_excss_to_canl_next_step(max_ncnls), str_cap(5),
     & tdqu_dn(max_ncnls), tempf_volwtr(max_n_res),
     & tempf_weircff(max_n_res,max_nostr_res),
     & tinflow(max_n_wcas,max_n_conv_cnl_wca), tivolwtr(max_n_res),
     & volwtr(max_n_res), wratio(max_ncnls),
     & total_small_afact(max_ncells)

      CHARACTER char_struc_name*6,
     & convey_canal_names(max_n_tot_conv_cnl_wcas)*5,
     & rlist_lvname(max_no_lvsp)*5, sname_s151ws*6,sname_s31ws*6,
     & struc_name_c9(10)*6, struc_name_meas(max_n_struc_meas)*6,
     & struc_name_s13a(10)*6

      LOGICAL l334rg, langel, langel_mx, lflowresid, lflwth, logdisg,
     & lpmps309, ls332mn, lws12ac, lws12bc, lws12cc, lws12dc, lwsdel

      REAL mean_land_surf_along_canal

      DOUBLE PRECISION crmin,stg_rmx,hw,tw,hws,tws,diff_stage,
     &tw_lo,floor,rmin_fc_stg,swl_land_surf,swl_max,stg_min_up,swl,
     &stage_m,twmax,stg_min,stg_max,stgdnc,dble13,hwstg,
     &rmax_ds_cnl_stg,swl_up_max,depth_below_target,hdiff,
     &avvolq,avvol_up,qpump,outflw_ds_cnl,avvoldn,avvol_dn,
     &aval_vol,cstor,avail_cstor,avvol_tot,avolsw,avvol_up_1,
     &avvoltoca1,avvolin,avvol_up_2,avalc51w,
     &rmin_stg_ds_end,stage_ups_s381
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      iflo=istr(n,ind1)
```

```
      char_struc_name=flnm(iflo)
      hw=hws
      tw=tws

C  GAPS IN L-5 TO TRANSFER WATER FROM HOLEY LAND TO WCA3

      gwdpth=(ells(nodehl)-h(nodehl))*s(nodehl)
      IF (pond(nodehl).gt.gwdpth) THEN
        hlstage=pond(nodehl)-gwdpth+ells(nodehl)
      ELSE
        hlstage=(pond(nodehl)-gwdpth)/s(nodehl)+ells(nodehl)
      END IF
      CALL bpts (iday, month, 10.0, 7, 1, 1, holey_sched)
C
      IF (ispec(n,ind1).ne.0) THEN
        IF (flnm(iflo).eq.strname_spec(26).or.flnm(iflo)
     &   .eq.strname_spec(27).or.flnm(iflo).eq.strname_spec(28)) THEN
         iwca=idn_wca_indx(is8bsn-6)
         IF (ihly_out_count.eq.0) THEN
           if (iroute_to_wca3a_frm_holy.eq.0) THEN
              holey_out_to_ds = 0.0
           endif
           total_outf_from_holy=0.0
         END IF
         ihly_out_count=ihly_out_count+1
         IF (hlyenv.eq.'TRUE ') THEN
           holy_stage=stage_outflw_wca(iholy)
         ELSE
           holy_stage=holey_sched
         END IF
         IF (hlyenv.eq.'FALSE') THEN
           IF (.not.dryseas.and.holy_oper.eq.'FLWTH') THEN
C
             deltah=chdep(n)-chdep(idncl)
             IF (deltah.gt.rminhd(n,ind1)) THEN
               cout=discoef(n,ind1,1,2)*(deltah-rminhd(n,ind1))**pwr(n,
     &          ind1,1,2)
             ELSE
               cout=0.0
             END IF
C
           ELSE
             IF (holy_oper.eq.'FLWTH') THEN
               IF (hlstage.ge.holy_stage+1.0.or.lflwth) THEN
                 lflwth=.true.
                 lflwth=.not.(hlstage.lt.holy_stage)
               END IF
               IF (lflwth) THEN
                 IF (chdep(n).gt.11.5) THEN
                   cout=discoef(n,ind1,1,1)*(chdep(n)-11.5)**pwr(n,
     &              ind1,1,1)
                 ELSE
                   cout=0.0
                 END IF
               ELSE
                 IF (chdep(n).gt.crmin(n,iseas)) THEN
                   cout=discoef(n,ind1,1,1)*(chdep(n)-crmin(n,iseas))**
```

```
     &               pwr(n,ind1,1,1)
                   ELSE
                     cout=0.0
                   END IF
                 END IF
               ELSE
                 IF (chdep(n).gt.crmin(n,iseas)) THEN
                   cout=discoef(n,ind1,1,1)*(chdep(n)-crmin(n,iseas))**
     &              pwr(n,ind1,1,1)
                 ELSE
                   cout=0.0
                 END IF
               END IF
             END IF
C
             IF (conv.eq.1.) THEN
               IF (iroute_to_wca3a_frm_holy.eq.0) THEN
                 qu(idncl)=qu(idncl)+cout*86400.
                 holey_out_to_ds=cout+holey_out_to_ds
               END IF
               kflo(iflo)=cout
               total_outf_from_holy=cout+total_outf_from_holy
             END IF
           ELSE
             IF (holy_outflow_opt.eq.'GRAV') THEN
               deltah=chdep(n)-chdep(idncl)
             ELSE
               deltah=chdep(n)-min(hlcnlds,chdep(idncl))
             END IF
             hlstage_out=avg_sim_wca_stg_out(iholy)
             IF (holy_oper.eq.'FLWTH') THEN
               IF (deltah.gt.rminhd(n,ind1)) THEN
                 coutmx=discoef(n,ind1,1,2)*(deltah-rminhd(n,ind1))**
     &            pwr(n,ind1,1,2)
               ELSE
                 coutmx=0.0
               END IF
               IF (hlstage_out.gt.holy_stage+offset_out(iholy,2)) THEN
                 cout=coutmx
               ELSE IF (hlstage_out.gt.holy_stage+offset_out(iholy,1))
     &          THEN
                 cout=1.0/(offset_out(iholy,2)-offset_out(iholy,1))*
     &            coutmx*(hlstage_out-holy_stage-offset_out(iholy,1))
               ELSE
                 cout=0.0
               END IF
             ELSE
               IF (chdep(n).gt.crmin(n,iseas)) THEN
                 cout=discoef(n,ind1,1,1)*(chdep(n)-crmin(n,iseas))**
     &            pwr(n,ind1,1,1)
               ELSE
                 cout=0.0
               END IF
             END IF
             IF (conv.eq.1) THEN
               IF (iroute_to_wca3a_frm_holy.eq.0) THEN
                 qu(idncl)=qu(idncl)+cout*86400.
```

```
                holey_out_to_ds=cout+holey_out_to_ds
              END IF
              kflo(iflo)=cout
              total_outf_from_holy=cout+total_outf_from_holy
c             print *,total_outf_from_holy,' HOLEY OUT'
            END IF
          END IF
C-----------------------------------------------------------
C             G200A (UP TO 250 CFS) OUTFLOW FROM HOLEYLAND
C-----------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(29)) THEN
          total_outf_from_holy_g200a=0.0
          IF (hlyenv.eq.'TRUE ') THEN
            holy_stage=stage_outflw_wca(iholy)
          ELSE
            holy_stage=holey_sched
          END IF
          IF (hlyenv.eq.'FALSE') THEN
            IF (holy_oper.eq.'FLWTH') THEN
              IF (hlstage.gt.holy_stage+0.25) THEN
                cout=discoef(n,ind1,1,1)
              ELSE IF (hlstage.gt.holy_stage) THEN
                cout=4.0*discoef(n,ind1,1,1)*(hlstage-holy_stage)
              ELSE
                cout=0.0
              END IF
            ELSE
              IF (chdep(n).gt.rmnstgs(n,ind1,iseas)+0.25) THEN
                cout=discoef(n,ind1,1,1)
              ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
                cout=4.0*discoef(n,ind1,1,1)*(chdep(n)-rmnstgs(n,ind1,
     &           iseas))
              ELSE
                cout=0.0
              END IF
            END IF
          ELSE
            hlstage_out=avg_sim_wca_stg_out(iholy)
            IF (holy_oper.eq.'FLWTH') THEN
              coutmx=discoef(n,ind1,1,1)
              IF (hlstage_out.gt.holy_stage+offset_out(iholy,2)) THEN
                cout=coutmx
              ELSE IF (hlstage_out.gt.holy_stage+offset_out(iholy,1))
     &         THEN
                cout=1.0/(offset_out(iholy,2)-offset_out(iholy,1))*
     &           coutmx*(hlstage_out-holy_stage-offset_out(iholy,1))
              ELSE
                cout=0.0
              END IF
            ELSE
              IF (chdep(n).gt.rmnstgs(n,ind1,iseas)+0.25) THEN
                cout=discoef(n,ind1,1,1)
              ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
                cout=4.0*discoef(n,ind1,1,1)*(chdep(n)-rmnstgs(n,ind1,
     &           iseas))
              ELSE
                cout=0.0
```

```
              END IF
            END IF
          END IF
          IF (conv.eq.1) THEN
            total_outf_from_holy_g200a=cout+total_outf_from_holy_g200a
            total_outf_from_holy=total_outf_from_holy_g200a
C             print *,total_outf_from_holy_g200a,' HOLEY OUT'
          END IF
C-----------------------------------------------------------------
C                    PUMP FROM C51W TO PROP L8 BPIT RES
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(30)) THEN
          cout=min(tot_avl_l8_bpit_res,max(discoef(n,ind1,1,1)-
     &     rl8runoff_to_bptres,0.0),tivolwtr(irsrcvr(n,ind1,1)))
          tivolwtr(irsrcvr(n,ind1,1))=tivolwtr(irsrcvr(n,ind1,1))-cout
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              irfsta=irsrcvr(n,ind1,1)
              IF (sfactor(irfsta).ge.sfactmin.or.ires_small_sim(irfsta)
     &         .eq.'NO ') THEN
                DO i=1,ncells_rec(n,ind1)
                  pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &             86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &             (nodrcv(n,ind1,i))-total_small_afact
     &             (nodrcv(n,ind1,i))))
                END DO
              ELSE
                DO kk=1,noresincell(nodrcv(n,ind1,1))
                  IF (irsrcvr(n,ind1,1).eq.ires_index_in_cell(nodrcv(n,
     &             ind1,1),kk)) THEN
                    IF (stage_res(nodrcv(n,ind1,1),kk)
     &                .gt.ells_in_res(nodrcv(n,ind1,1),kk)) THEN
                      pond_in_res(nodrcv(n,ind1,1),kk)=
     &                 pond_in_res(nodrcv(n,ind1,1),kk)+cout*86400./
     &                 (sfactor(irsrcvr(n,ind1,1))*gdar)
                      stage_res(nodrcv(n,ind1,1),kk)=
     &                 ells_in_res(nodrcv(n,ind1,1),kk)+
     &                 pond_in_res(nodrcv(n,ind1,1),kk)
                    ELSE
                      dpth_below_ls_res=(ells_in_res(nodrcv(n,ind1,1),
     &                 kk)-stage_res(nodrcv(n,ind1,1),kk))*s(nodrcv(n,
     &                 ind1,1))
                      dpth_of_inflw=cout*86400./(sfactor(irsrcvr(n,ind1,
     &                 1))*gdar)
                      stage_res(nodrcv(n,ind1,1),kk)=stage_res(nodrcv(n,
     &                 ind1,1),kk)+amin1(dpth_of_inflw,
     &                 dpth_below_ls_res)/s(nodrcv(n,ind1,1))+
     &                 amax1(dpth_of_inflw-dpth_below_ls_res,0.0)
                      pond_in_res(nodrcv(n,ind1,1),kk)=
     &                 amax1(stage_res(nodrcv(n,ind1,1),kk)-
     &                 ells_in_res(nodrcv(n,ind1,1),kk),0.0)
                    END IF
                    res_struc_inflw(irsrcvr(n,ind1,1))=
     &               res_struc_inflw(irsrcvr(n,ind1,1))+cout*1.9835
                  END IF
```

```
                END DO
              END IF
            END IF
            kflo(iflo)=cout
C            print *,tot_avl_l8_bpit_res_lwd1,tot_avl_l8_bpit_res
C    $         ,DISCOEF(N,IND1,1,1),rl8runoff_to_bptres
C    $         ,tot_avl_l8_bpit_res_m1q,VOLWTR(irsrcvr(n,ind1,1))
C    $         ,cout
            volwtr(irsrcvr(n,ind1,1))=volwtr(irsrcvr(n,ind1,1))-cout
          END IF
C
C-----------------------------------------------------------------
C                    NSIMP DIST PUMPS
C-----------------------------------------------------------------
        ELSE IF (char_struc_name(1:5).eq.'NSIMP') THEN
          cout=0.0
          CALL match ('HLSB ', 1, 5, cnm, nch, 5, ihlsb)
          IF ((chdep(ihlsb).gt.rmnstgs(ihlsb,1,iseas)+0.25.or.
     &      canl_rf_avg(ihlsb).gt.3.80/14. .and. irceive(n,ind1)
     &      .gt. 1) .or. irceive(n,ind1) .eq. 1) THEN
             cout=capacfl(imgt,1)
          END IF
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-------------------------------------------------------------------
C  CULVERTS FROM ACMEB BASIN TO C51W AND S-155A
C-------------------------------------------------------------------
         ELSE IF (flnm(iflo) .eq. strname_spec(95)) THEN
           DO nnn = 1,nch
              if (cnm(nnn) .eq. 'ACMEA') THEN
                 n_cnl_indx_ac = nnn
                 go to 5678
              endif
           ENDDO
 5678      tw = chdep(n_cnl_indx_ac)
c
           if (hw .gt. tw) then
              struc_capac0 = discoef(n,ind1,1,1)*(hw-tw)
     &        ** pwr(n,ind1,1,1)
           else
              struc_capac0 = 0.0
           endif
           if (hw .gt. rmnstgs(n,ind1,iseas)+0.01) then
              struc_capac = struc_capac0
           elseif (hw .gt. rmnstgs(n,ind1,iseas)) then
              struc_capac = 100.0*struc_capac0
```

```
     &         *(hw-rmnstgs(n,ind1,iseas))
           else
              struc_capac = 0.0
           endif
c
           cout = struc_capac
           if (conv.eq.1.) then
              qu(idncl) = qu(idncl) + cout*86400.
              kflo(iflo) = cout
              if (iroute_acme_fc_to_sta1e_opt .eq. 0) then
                out_flow_east0 = out_flow_east0 + cout
c               print *,hw,tw,rmnstgs(n,ind1,iseas)
c    &            ,iroute_acme_fc_to_sta1e_opt
c    &            ,out_flow_east0,cout,struc_capac0,'  ACMEO'
              endif
           endif
C
C-----------------------------------------------------------------
C                            L31N PUMP STATION
C-----------------------------------------------------------------
C
        ELSE IF (char_struc_name(1:5).eq.'S356 ') THEN
          IF (kflo(ifls334).ne.0.or.dqu(l31s).gt.0.0) THEN
            capacfl(iseas,1)=0.0
          ELSE
            capacfl(iseas,1)=discoef(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &       month)
          END IF
          diffs=rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas)
          IF (stg_trig_fc_ops(n,ind1).lt.rmnstgs(n,ind1,iseas)+diffs
     &    .and.stg_trig_fc_ops(n,ind1).gt.rmnstgs(n,ind1,iseas)) THEN
            qpump=(stg_trig_fc_ops(n,ind1)-rmnstgs(n,ind1,iseas))*
     &     (1.0/diffs)*discoef(n,ind1,1,1)*frac_struc_cap(n,ind1,month)
          ELSE IF (stg_trig_fc_ops(n,ind1).ge.rmnstgs(n,ind1,iseas)
     &    +diffs) THEN
            qpump=discoef(n,ind1,1,1)*frac_struc_cap(n,ind1,month)
          ELSE
            qpump=0.0
          END IF
          diff_stg_f_dnend = stg_trig_fc_ops(n,ind1)-chdep(n)
          rmin_stg_ds_end = rmnstgs(n,ind1,iseas)
     &    - diff_stg_f_dnend
          avvol_up=(hw-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+cgseep(n,
     &     rmin_stg_ds_end,0,0.)+tovflc(n,rmin_stg_ds_end,0,
     &     0.,iter,1)+qu(n)
          avvol_up=max(avvol_up,0.0)/86400.
          IF (iseep_lim_s356_opt.ne.0) THEN
            avvol_up_s=max(qu(n)-kflo(ifls335)*86400.-kflo(ifls334)*
     &       86400.-s334_fc*86400.,0.0)
            avvol_up=min(avvol_up_s/86400.,avvol_up)
          END IF
          qpumpt=capacfl(iseas,1)
          cout=min(qpumpt,discoef(n,ind1,1,1),avvol_up,qpump)
c         print *,qpumpt,discoef(n,ind1,1,1),avvol_up,qpump,cout,' S356'
c    &    ,avvol_up_s/86400.,hw,rmnstgs(n,ind1,iseas),kflo(ifls335)
c    &    ,kflo(ifls334),qu(n)/86400.,iseep_lim_s356_opt
          IF (irceive(n,ind1).eq.1) THEN
```

```
        IF (opt_mrp_to_enp) THEN
          IF (chdep(idncl).gt.cnlstg(n,ind1,1)-0.25) THEN
            frac_max_disc_c=(cnlstg(n,ind1,1)-chdep(idncl))/0.25
            frac_max_disc_c=amax1(frac_max_disc_c,0.0)
          ELSE
            frac_max_disc_c=1.0
          END IF
        ELSE
          frac_max_disc_c=1.0
        END IF
      ELSE
        frac_max_disc_c=1.0
        IF (stage3b.gt.8.5.or.nint(rmnstg_dual_ops0(n,ind1)).eq.99)
&        THEN
          cout=0.0
        END IF
      END IF
      dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
      IF (pond(node_g3273).gt.dpthgw) THEN
        stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
      ELSE
        stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
&        ells(node_g3273)
      END IF
      IF (opt_mrp_to_enp.and.irceive(n,ind1).eq.1) THEN
        DO ncal = 1,nch
          DO i=1,nstrctr(ncal)
            IF (tmpname(ncal,i).eq.'S333  ') THEN
               istruc_s333=i
               incl_s333 = ncal
            ENDIF
          END DO
        ENDDO
        IF (stage_g3273.gt.stg_trigger_fl(incl_s333,istruc_s333
&       ,1)) THEN
          frac_max_disc_gc=0.0
        ELSE
          frac_max_disc_gc=1.0
        END IF
      ELSE
        frac_max_disc_gc=1.0
      END IF
      cout=cout*amin1(frac_max_disc_gc,frac_max_disc_c)
      IF (conv.eq.1) THEN
        IF (irceive(n,ind1).eq.1) THEN
          qu(idncl)=qu(idncl)+cout*86400.
          s356_to_l29=cout
          s356_to_cell=0.0
        ELSE
          DO i=1,ncells_rec(n,ind1)
            pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
&            86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
&            (nodrcv(n,ind1,i))-total_small_afact
&            (nodrcv(n,ind1,i))))
            s356_to_cell=cout
            s356_to_l29=0.0
          END DO
```

```
            END IF
            kflo(iflo)=cout
          END IF
C
C-----------------------------------------------------------------
C                   S-356 FOR IOP
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.'S356IO') THEN
          IF (nint(dqu(l31s)).gt.0.or.nint(s337fc)
     &     .gt.0.or.nint(kflo(ifls337)).gt.0) THEN
            capacfl(iseas,1)=0.0
          ELSE
            capacfl(iseas,1)=discoef(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &       month)
          END IF
          avvol_up=kflo(ifls335)
          qpumpt=capacfl(iseas,1)
          stage3b=trig_wca_stg(iwca3b)
          IF ((stage3b.lt.8.5.and.irceive(n,ind1).ne.1).or.irceive(n,
     &     ind1).eq.1) THEN
            IF (opt_mrp_to_enp.and.irceive(n,ind1).eq.1) THEN
              IF (chdep(idncl).gt.cnlstg(n,ind1,1)-0.25) THEN
                frac_max_disc_c=(cnlstg(n,ind1,1)-chdep(idncl))/0.25
                frac_max_disc_c=amax1(frac_max_disc_c,0.0)
              ELSE
                frac_max_disc_c=1.0
              END IF
            ELSE
              frac_max_disc_c=1.0
            END IF
            dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
            IF (pond(node_g3273).gt.dpthgw) THEN
              stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
            ELSE
              stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
     &         ells(node_g3273)
            END IF
            IF (opt_mrp_to_enp.and.irceive(n,ind1).eq.1) THEN
              DO i=1,nstrctr(ica3)
                IF (tmpname(ica3,i).eq.'S333  ') istruc_s333=i
              END DO
              IF (stage_g3273.gt.stg_trigger_fl(ica3,istruc_s333,1))
     &         THEN
                frac_max_disc_gc=0.0
              ELSE
                frac_max_disc_gc=1.0
              END IF
            ELSE
              frac_max_disc_gc=1.0
            END IF
            cout=min(qpumpt,avvol_up)*amin1(frac_max_disc_c,
     &       frac_max_disc_gc)
          ELSE
            cout=0.0
          END IF
          IF (conv.eq.1) THEN
```

```
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              s356_to_cell=0.0
              s356_to_l29=cout
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
              s356_to_cell=cout
              s356_to_l29=0.0
            END IF
            kflo(iflo)=cout
          END IF
C
C-------------------------------------------------------------
C                   PUMPAGE OF C6W WATER TO SO WRA
C-------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.'C6WWRA') THEN
          CALL match ('LBLTN', 1, 5, cnm, nch+1, 5, ibltn)
          IF (ibltn.eq.0) ibltn=nch+1
          IF (ibltn.lt.nch) THEN
            IF (nint(dqu(ibltn)).eq.0) THEN
              cout=capacfl(imgt,1)
            ELSE
              cout=0.0
            END IF
          ELSE
            cout=capacfl(imgt,1)
          END IF
          IF (irceive(n,ind1) .ne. 1) then
             irfsta=irsrcvr(n,ind1,1)
             cout = min(cout,volwtr(irfsta))
          ENDIF
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              IF (sfactor(irfsta).ge.sfactmin.or.ires_small_sim(irfsta)
     &         .eq.'NO ') THEN
                DO i=1,ncells_rec(n,ind1)
                  pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &             86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &             (nodrcv(n,ind1,i))-total_small_afact
     &             (nodrcv(n,ind1,i))))
                END DO
              ELSE
                DO kk=1,noresincell(nodrcv(n,ind1,1))
                  IF (irsrcvr(n,ind1,1).eq.ires_index_in_cell(nodrcv(n,
     &             ind1,1),kk)) THEN
                   IF (stage_res(nodrcv(n,ind1,1),kk)
     &                .gt.ells_in_res(nodrcv(n,ind1,1),kk)) THEN
                    pond_in_res(nodrcv(n,ind1,1),kk)=
     &                 pond_in_res(nodrcv(n,ind1,1),kk)+cout*86400./
     &                 (sfactor(irsrcvr(n,ind1,1))*gdar)
```

```
                    stage_res(nodrcv(n,ind1,1),kk)=
     &               ells_in_res(nodrcv(n,ind1,1),kk)+
     &               pond_in_res(nodrcv(n,ind1,1),kk)
                  ELSE
                    dpth_below_ls_res=(ells_in_res(nodrcv(n,ind1,1),
     &               kk)-stage_res(nodrcv(n,ind1,1),kk))*s(nodrcv(n,
     &               ind1,1))
                    dpth_of_inflw=cout*86400./(sfactor(irsrcvr(n,ind1,
     &               1))*gdar)
                    stage_res(nodrcv(n,ind1,1),kk)=stage_res(nodrcv(n,
     &               ind1,1),kk)+amin1(dpth_of_inflw,
     &               dpth_below_ls_res)/s(nodrcv(n,ind1,1))+
     &               amax1(dpth_of_inflw-dpth_below_ls_res,0.0)
                    pond_in_res(nodrcv(n,ind1,1),kk)=
     &               amax1(stage_res(nodrcv(n,ind1,1),kk)-
     &               ells_in_res(nodrcv(n,ind1,1),kk),0.0)
                  END IF
                END IF
              END DO
            END IF
            volwtr(irfsta)=volwtr(irfsta)-cout
          END IF
          kflo(iflo)=cout
        END IF
C
C-------------------------------------------------------------
C                   PUMPAGE OF C6W WATER TO SO WRA
C-------------------------------------------------------------
      ELSE IF (flnm(iflo).eq.'C6DIVS') THEN
        IF (hw.gt.rmxstg(n,ind1,iseas)) THEN
          coutfl=capacfl(imgt,1)
        ELSE IF (hw.gt.rmnstgs(n,ind1,iseas)) THEN
          pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**3*
     &     (hw-rmnstgs(n,ind1,iseas))**3
          coutfl=pctgo*capacfl(imgt,1)
        ELSE
          coutfl=0.0
        END IF
        CALL match ('LBLTN', 1, 5, cnm, nch+1, 5, ibltn)
        IF (ibltn.eq.0) ibltn=nch+1
        IF (ibltn.lt.nch) THEN
          IF (nint(dqu(ibltn)).eq.0) THEN
            cout=coutfl
          ELSE
            cout=0.0
          END IF
        ELSE
          cout=coutfl
        END IF
        IF (conv.eq.1) THEN
          IF (irceive(n,ind1).eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
          ELSE
            DO i=1,ncells_rec(n,ind1)
              pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &         86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &         (nodrcv(n,ind1,i))-total_small_afact
```

```
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------------------
C              FLOW FRM ACMERS TO C51W
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.'ACBC51') THEN
          CALL match ('ACMERS', 1, 6, resname, ntotres+1, 6, iacmrs)
          IF (flow_to_acmres_frm_basin.eq.weircff(iacmrs,1,1)
     &     .or.nint(volwtr(iacmrs)).eq.0) THEN
            cout=capacfl(imgt,1)
          ELSE
            cout=0.0
          END IF
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C
C-----------------------------------------------------------------
C             WATER SUPPLY FROM L-8 TO M-CANAL
C                 ORIGINATING FROM WCA-1
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(33)) THEN
          hdiff=abs(chdep(n)-chdep(idncl))
          IF (chdep(n).lt.chdep(idncl)) THEN
            IF (chdep(n).gt.10.2) THEN
              capac=7.15*(6.0-hdiff)+182.0
            ELSE IF (chdep(n).ge.9.5) THEN
              IF (hdiff.gt.11.5) THEN
                capac1=136.0-10.29*(hdiff-11.5)
              ELSE IF (hdiff.gt.4.0) THEN
                capac1=136.0+4.53*(11.5-hdiff)
              ELSE
                capac1=170.0
              END IF
              IF (chdep(n).gt.10.15) THEN
                capac0 = 7.15*(6.0-hdiff)+182.0
                capac = 20.0*(chdep(n)-10.15)*(capac0-capac1)
     &          + capac1
              ELSE
                capac = capac1
              ENDIF
            ELSE IF (chdep(n).gt.9.1) THEN
              capac=150.0*(chdep(n)-9.1)/0.4
```

```
          ELSE
            capac=0.0
          END IF
        ELSE
          IF (chdep(n).ge.9.5) THEN
            capac=170.0
          ELSE IF (chdep(n).gt.9.1) THEN
            capac=170.0*(chdep(n)-9.1)/0.4
          ELSE
            capac=0.0
          END IF
        END IF
        capac=capac*facts2mc
        IF (stagelo.lt.sched(iday,month)) THEN
          twsneed=dqu(ilkmg)+dmndagmc
          capac=amin1(capac,twsneed,50.0)
        END IF
C
        gwdpth=(ells(iwpbcat)-h(iwpbcat))*s(iwpbcat)
        IF (pond(iwpbcat).gt.gwdpth) THEN
          stage=pond(iwpbcat)-gwdpth+ells(iwpbcat)
        ELSE
          stage=(pond(iwpbcat)-gwdpth)/s(iwpbcat)+ells(iwpbcat)
        END IF
C
        IF (chdep(n).gt.rmin_l8stg_flow_to_mcnl(iseas)) THEN
          stg_rmx=rmax_wpbcat_for_inflw
          IF (stage.lt.rmax_wpbcat_for_inflw) THEN
            dishgdn=0.0
            tot_pump=0.0
            DO ns=1,nstrctr(idncl)
              dishgdn1=strout(stg_rmx,n,idncl,ns,0,crmin,volwtr,
     &         prevqu,rain,iter,tot_pump,jday,tempf_volwtr,
     &         tempf_weircff,0)
              dishgdn=dishgdn+dishgdn1
            END DO
            avvoldn=(stg_rmx-chdep(idncl))*cl(idncl)*width(idncl)-
     &       cgseep(idncl,stg_rmx,0,0.)-tovflc(idncl,stg_rmx,0,0.,iter
     &       ,2)-qu(idncl)+dishgdn*86400.
            avvoldn=max(avvoldn/86400.,0.0)
            IF (chdep(n).gt.rmin_l8stg_flow_to_mcnl(iseas)+0.25) THEN
              capac_out=capac
            ELSE
              capac_out=4.0*capac*(chdep(n)-
     &         rmin_l8stg_flow_to_mcnl(iseas))
            END IF
            cout=min(capac_out,avvoldn,capac_mcnl-
     &       total_flow_to_mcnl)
          ELSE
            cout=0.0
          END IF
        ELSE
          stg_rmx=crmin(idncl,iseas)
          dishgdn=0.0
          tot_pump=0.0
          DO ns=1,nstrctr(idncl)
            dishgdn1=strout(stg_rmx,n,idncl,ns,0,crmin,volwtr,prevqu,
```

```
     &         rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
              dishgdn=dishgdn+dishgdn1
            END DO
            avvoldn=(stg_rmx-chdep(idncl))*cl(idncl)*width(idncl)-
     &       cgseep(idncl,stg_rmx,0,0.)-tovflc(idncl,stg_rmx,0,0.,iter
     &       ,2)-qu(idncl)+dishgdn*86400.
            avvoldn=max(avvoldn/86400.,0.0)
            cout=min(capac,avvoldn)
          END IF
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
            total_flow_to_mcnl=total_flow_to_mcnl+cout
          END IF
C-------------------------------------------
C        M1 LOWER BASIN TO MCNL
C-------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(34)) THEN
C
C  upsteam avail volume
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up,0.0)/86400.
C
C  downstream avail storage
          dishgdn=0.0
          tot_pump=0.0
          stg_rmx=rmax_wpbcat_for_inflw
          DO ns=1,nstrctr(idncl)
            dishgdn1=strout(stg_rmx,n,idncl,ns,0,crmin,volwtr,prevqu,
     &       rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
            dishgdn=dishgdn+dishgdn1
          END DO
          avvoldn=(stg_rmx-chdep(idncl))*cl(idncl)*width(idncl)-
     &     cgseep(idncl,stg_rmx,0,0.)-tovflc(idncl,stg_rmx,0,0.,iter
     &     ,2)-qu(idncl)+dishgdn*86400.
          avvoldn=max(avvoldn/86400.,0.0)
C
C  calculate outflow
          cout=min(avvol_up,avvoldn,discoef(n,ind1,1,1),capac_mcnl-
     &     total_flow_to_mcnl)
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            IF (idncl.lt.n) THEN
              rloc_excss_to_canl_next_step(idncl)=
     &         rloc_excss_to_canl_next_step(idncl)+cout*86400.
            END IF
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
            total_flow_to_mcnl=total_flow_to_mcnl+cout
          END IF
C-------------------------------------------
C        M1 UPPER BASIN TO MCNL
C-------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(35)) THEN
```

```
C
C  upsteam avail volume
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up,0.0)/86400.
C
C  downstream avail storage
          dishgdn=0.0
          tot_pump=0.0
          stg_rmx=rmax_wpbcat_for_inflw
          DO ns=1,nstrctr(idncl)
            dishgdn1=strout(stg_rmx,n,idncl,ns,0,crmin,volwtr,prevqu,
     &       rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
            dishgdn=dishgdn+dishgdn1
          END DO
          avvoldn=(stg_rmx-chdep(idncl))*cl(idncl)*width(idncl)-
     &     cgseep(idncl,stg_rmx,0,0.)-tovflc(idncl,stg_rmx,0,0.,iter
     &     ,2)-qu(idncl)+dishgdn*86400.
          avvoldn=max(avvoldn/86400.,0.0)
C
C  calculate outflow
          cout=min(avvol_up,avvoldn,discoef(n,ind1,1,1),capac_mcnl-
     &     total_flow_to_mcnl)
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
            total_flow_to_mcnl=total_flow_to_mcnl+cout
          END IF

C-----------------------------------------------------------------
C                STRUCTURE TO ALLOW BACK FLOW BY
C                 GRAVITY FROM M-CANAL TO L-8
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(36)) THEN
          IF (hw.gt.rmnstgs(n,ind1,iseas)+1.0) THEN
            cout=discoef(n,ind1,1,1)
          ELSE IF (hw.gt.rmnstgs(n,ind1,iseas)) THEN
     &
            cout=1.0*discoef(n,ind1,1,1)*(hw-
     &       rmnstgs(n,ind1,iseas))
          ELSE
            cout=0.0
          END IF
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
          END IF
C
C---------------------------------------------------------------
C             BACKFLOW FROM L-8 INTO LAKE OKEECHOBEE
C---------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(37)) THEN
C
          IF (lok_split_option.eq.1) THEN
```

```
            tw_lo=max(stagelo,stagelo_res)
          ELSE
            tw_lo=stagelo
          END IF
          IF (hw.gt.tw_lo) THEN
C
            avvol_up=(hw-tw_lo)*cl(n)*width(n)+
     &       cgseep(n,tw_lo-hdc(n,iseas),0,0.)+tovflc(n,tw_lo-hdc(n,
     &       iseas),0,0.,iter,1)+qu(n)
            avvol_up=max(avvol_up/86400.,0.0)
            cout=min(avvol_up,capacfl(1,1))
          ELSE
            cout=0.0
          END IF
          IF (conv.eq.1) THEN
            kflo(iflo)=-cout+kflo(iflo)
            IF (lok_split_option.eq.0) THEN
              finstor=finstor+cout*1.9835
              stagelo=stg2sto(finstor,1)
            ELSE
              finstor_res=finstor_res+cout*1.9835
              stagelo_res=stg2stores(finstor_res,1)
            END IF
          END IF
C
C-----------------------------------------------------------------
C             WATER SUPPLY FROM CATCHMENT AREA TO  EASTERN
C               REACH OF M - CANAL
C----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(38)) THEN
          avvol_up=(chdep(n)-chdep(idncl))*cl(n)*width(n)+cgseep(n,
     &     chdep(idncl),0,0.)+tovflc(n,chdep(idncl),0,0.,iter,1)+qu(n)
          IF (chdep(n).gt.14.5) THEN
            rmxdschg=discoef(n,ind1,1,1)
          ELSE IF (chdep(n).gt.12.5) THEN
            rmxdschg=12.5*(chdep(n)-12.5)+25.0
          ELSE
            rmxdschg=25.0
          END IF
          avvol_up=max(avvol_up/86400.,0.0)
          cout=min(dqu(ilkmg),avvol_up,rmxdschg)
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
          END IF
C----------------------------------------------------------------
C              S142 BACK TO WCA3A
C----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(39)) THEN
          IF (hw.gt.tw) THEN
            s142cap=discoef(n,ind1,1,1)*(hw-tw)**pwr(n,ind1,1,1)
          ELSE
            s142cap=0.0
          END IF
          CALL match ('S34   ', 1, 6, struc_name_meas, nflpts, 6, ims34)
          flimit=tflg123+amax1(kflo(ifls143)+kflo(ifls141)-kflo(ifls34),
```

```
     &     0.0)
          IF (ims34.eq.0) THEN
            cout=amin1(s142cap,flimit)
          ELSE
            cout=flimit
          END IF
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            kflo(iflo)=kflo(iflo)-cout
          END IF
C-----------------------------------------------------------------
C              S-151 WATER SUPPLY
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(40)) THEN
C-----------------------------------------------------------------
          CALL match ('S337  ', 1, 6, struc_name_meas, nflpts, 6,
     &     imatch_s337)
          CALL match ('C6   ', 1, 5, cnm, nch+1, 5, ic6)
          IF (imatch_s337.ne.0) THEN
            cout=kflo(ifls337)+dqu(ic6)+kflo(iflo)
            IF (conv.eq.1.) THEN
              kflo(iflo)=cout
              upstream_inflow(idncl)=upstream_inflow(idncl)+
     &         (kflo(ifls337)+dqu(ic6))*86400.
              wss151=kflo(ifls337)+dqu(ic6)+wss151
              add_ws_s151=kflo(ifls337)+dqu(ic6)
            END IF
            GO TO 10
          END IF
          IF (lwslo_floor(iwca3a)) THEN
            aval_vol=0.0
          ELSE
            iconvey_c=iconvey_c_s151
            floor=floor_elev(iwca3a,iconvey_c)
            cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &       cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c)
     &       )
            aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)
     &       +tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
          END IF
C     AVVOL_TOT = AMAX1(AVAL_VOL,0.0)/86400. + TINFLOW(IWCA3A,iconvey_c)
C     AVVOL_TOT = AMAX1(AVVOL_TOT,0.0)
          tinflow_rev=tinflow(iwca3a,iconvey_c)-
     &     outflow_from_3a_temp(iconvey_c)
          IF (iprorte(iwca3a).ne.1) tinflow_rev=tinflow(iwca3a,
     &     iconvey_c)-outflow_from_3a_temp(iconvey_c)-total_flow_s12
          avvolws(iwca3a,iconvey_c)=max(aval_vol,0.0)+tinflow_rev*
     &     86400.
          avvolws(iwca3a,iconvey_c)=max(avvolws(iwca3a,iconvey_c),0.0)
          avvolws(iwca3a,iconvey_c)=avvolws(iwca3a,iconvey_c)/86400.
C     if (nint(TNEED(IWCA3A,iconvey_c)) .GT. 0) THEN
C       RATIO = AVVOL_TOT / TNEED(IWCA3A,iconvey_c)
C     else
C       RATIO = 1.0
C     endif
          hwstg=chdep(n)+0.4*hdc(n,iseas)
          IF (hwstg.gt.chdep(idncl)) THEN
```

```
            s151cap=discoef(n,ind1,1,1)*(hwstg-chdep(idncl))**pwr(n,
     &       ind1,1,1)
          ELSE
            s151cap=0.0
          END IF
          rem_s151cap=amax1(s151cap-kflo(ifls151),0.0)
C
C -------------
C  DETERMINE WATER SUPPLY RELEASES THROUGH S-151 AND S-337 (GATED SPILLW
C -------------
C
          wsneed_s151=0.0
          DO i=1,no_ds_cnl_ws(n,ind1)
            wsneed_s151=wsneed_s151+tdqu_dn(idncnl_ws(n,ind1,i))
          END DO
          rem_wsneed_s151=wsneed_s151-kflo(ifls151)
          rem_wsneed_s151=amax1(rem_wsneed_s151,0.0)
          IF (nint(rem_wsneed_s151).gt.0) THEN
            ratio=avvolws(iwca3a,iconvey_c)/(rem_wsneed_s151+frf)
          ELSE
            ratio=1.0
          END IF
C
          IF (nint(rem_wsneed_s151).gt.0) THEN
            avvol_upstream=ratio*rem_wsneed_s151
            ratio=amin1(ratio,1.0)
C
            IF (iprorte(iwca3a).eq.1) THEN
C
C   DETERMINE WATER SUPPLY RELEASE THRU S-151
C
              wsdel=amax1(rem_wsneed_s151*ratio,0.0)
              cout=amin1(wsdel,rem_s151cap)*ratflw*
     &         frac_del_frm_wca(iwca3a)
            ELSE
              avvol_upstream=avvolws(iwca3a,iconvey_c)
              rem_need=amax1(wsneed_s151-kflo(ifls151),0.0)
              cout=amin1(rem_need,rem_s151cap,avvolws(iwca3a,iconvey_c))
     &         *ratflw*frac_del_frm_wca(iwca3a)
            END IF
          ELSE
            cout=0.0
            wsneed=0.0
            avvol_upstream=avvolws(iwca3a,iconvey_c)
          END IF
          IF (conv.eq.1) THEN
            kflo(iflo)=cout+kflo(iflo)
            wss151=cout+wss151
            add_ws_s151=cout
            qu(idncl)=qu(idncl)+cout*86400.
            dqu(idncl)=dqu(idncl)-amin1(wss151,dqu(idncl))
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            avvolws(iwca3a,iconvey_c)=avvolws(iwca3a,iconvey_c)-cout
            rem_wsneed_s151=rem_wsneed_s151-cout
            sname_s151ws='s151ws'
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,sname_s151ws,
```

```
     &         chdep(n),chdep(idncl),s151cap,avvol_upstream,wss151,
     &         wsneed_s151
            END IF
          END IF
   10     CONTINUE

C-------------------------------------------------------------
C                S-31
C-------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(41)) THEN
C -------------
C  WATER SUPPLY TO C-6
C -------------
          IF (chdep(n).gt.chdep(idncl)) THEN
            s31cap=discoef(n,ind1,1,1)*(chdep(n)-chdep(idncl))**pwr(n,
     &       ind1,1,1)
          ELSE
            s31cap=0.0
          END IF
          IF (n.eq.intcnl(idn_wca_indx(is8bsn-6),
     &     iconvey_canal_no(is150bws-6))) THEN
            iconvey_c=iconvey_canal_no(is150bws-6)
            IF (lwslo_floor(idn_wca_indx(is8bsn-6))) THEN
              aval_vol=tinflow(iwca3a,iconvey_c)-
     &         outflow_from_3a_temp(iconvey_c)
            ELSE
              aval_vol=tempvol
            END IF
          ELSE
            aval_vol=tempvol
          END IF
          tempvol=aval_vol
          rem_cap_s31=amax1(s31cap-kflo(ifls31),0.0)
          avvol_upstream_c6=tempvol
          IF (dqu(idncl).gt.0.0) THEN
            IF (nprty(n,ind1).ne.0) THEN
              rem_need_s31=amax1(dqu(idncl)-kflo(ifls31),0.0)
              cout=amin1(rem_need_s31,tempvol,rem_cap_s31)*
     &         frac_del_frm_wca(iwca3a)
            ELSE
              volrat=tempvol/tdmnd
              wratio(n)=amin1(volrat,1.0)
              wsneed=dqu(idncl)*wratio(n)-kflo(ifls31)
              wsneed=amax1(wsneed,0.0)
              cout=amin1(wsneed,rem_cap_s31)*frac_del_frm_wca(iwca3a)
              tdmnd=tdmnd-dqu(idncl)
            END IF
          ELSE
            cout=bseflw
          END IF
          tempvol=tempvol-cout
          IF (conv.eq.1) THEN
            kflo(iflo)=cout+kflo(iflo)
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            wss31=cout+wss31
            IF (n.eq.intcnl(idn_wca_indx(is8bsn-6),
```

```
     &       iconvey_canal_no(is150bws-6))) THEN
              outflow_from_3a(iconvey_canal_no(is150bws-6))=
     &         outflow_from_3a(iconvey_canal_no(is150bws-6))+cout
              outflow_from_3a_temp(iconvey_canal_no(is150bws-6))=
     &         outflow_from_3a(iconvey_canal_no(is150bws-6))
            END IF
            sname_s31ws='S31WS '
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,sname_s31ws,chdep(n)
     &         ,chdep(idncl),s31cap,avvol_upstream_c6,wss31,dqu(idncl)
            END IF
            dqu(idncl)=dqu(idncl)-amin1(wss31,dqu(idncl))
          ELSE
            IF (n.eq.intcnl(idn_wca_indx(is8bsn-6),
     &       iconvey_canal_no(is150bws-6))) THEN
              outflow_from_3a_temp(iconvey_canal_no(is150bws-6))=
     &         outflow_from_3a(iconvey_canal_no(is150bws-6))+cout
            END IF
          END IF
C-----------------------------------------------------------------
C          S-381BK EQUALIZER FLOW (CONJUCTION WITH S381 BALLOON
C             DIVIDE STRUCTURE)
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq. 'S381BK') THEN
            do nn = 1,nstrctr(n)
               if (tmpname(n,nn).eq.'S9  ') then
                  is_s9 = nn
               endif
               if (tmpname(n,nn).eq.'S9A ') then
                  is_s9a = nn
               endif
            enddo
            do nn = 1,nstrctr(ic11w)
               if (tmpname(ic11w,nn).eq.'S381 ') then
                 is_s381 = nn
               endif
            enddo
            stage_ups_s381 = chdep(ic11w) + hdc(ic11w,iseas)
            avvol_ups = (chdep(n)-stage_ups_s381)
     &      * cl(n)*width(n) + qu(n) + cgseep(n,stage_ups_s381
     &      ,0,0.) + tovflc(n,stage_ups_s381,0,0.,iter,1)
            avvol_ups = amax1(avvol_ups,0.0)/86400.
            if (nint(dqu(ic11w)).eq.0
     &       .and. .not. lpump(n,is_s9,1) .and. outf_s9a
     &       .ge.discoef(n,is_s9a,1,1)-2.0.and.lgrav(ic11w,is_s381,1))
     &       then
               if (chdep(n) .gt. stage_ups_s381+0.50) then
                  capac_eq = capacfl(imgt,1)
               else
                  capac_eq = 2.0*max(chdep(n)-stage_ups_s381
     &             ,0.0)*capacfl(imgt,1)
               endif
               capac_eq1 = 0.5*capacfl(imgt,1)*(outf_s9a-discoef
     &           (n,is_s9a,1,1) + 2.0)
               cout = amin1(capac_eq,avvol_ups,capac_eq1)
            else
               cout = 0.0
```

```
            endif
            IF (conv.eq.1.) THEN
c             print *,stage_ups_s381,chdep(n),avvol_ups,cout
c    &        ,capac_eq,lpump(n,is_s9,1),'  S381BK '
c    &        ,outf_s9a,capacfl(imgt,1),imgt
c    &        ,is_s381,lgrav(ic11w,is_s381,1)
              IF (irceive(n,ind1).eq.1) THEN
                qu(idncl)=qu(idncl)+cout*86400.
                upstream_inflow(idncl)=upstream_inflow(idncl)+cout
     &          *86400.
              ELSE
                DO i=1,ncells_rec(n,ind1)
                 pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
                END DO
              END IF
              kflo(iflo) = cout
            ENDIF
C-----------------------------------------------------------------
C              S-9 WATER SUPPLY
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(8)) THEN
C
          IF (lwslo_floor(iwca3a)) THEN
            aval_vol=0.0
          ELSE
            iconvey_c=iconvey_c_s151
            floor=floor_elev(iwca3a,iconvey_c)
            cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &       cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c)
     &       )
            aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)
     &       +tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
          END IF
          avvol_tot=max(aval_vol,0.0)+tinflow(iwca3a,iconvey_c)
          tinflow_rev=tinflow(iwca3a,iconvey_c)-
     &     outflow_from_3a_temp(iconvey_c)
          avvolws(iwca3a,iconvey_c)=max(aval_vol,0.0)+tinflow_rev*
     &     86400.
          avvolws(iwca3a,iconvey_c)=max(avvolws(iwca3a,iconvey_c),0.0)
          avvolws(iwca3a,iconvey_c)=avvolws(iwca3a,iconvey_c)/86400.
          IF (nint(tneed(iwca3a,iconvey_c)).gt.0) THEN
            ratio=avvol_tot/tneed(iwca3a,iconvey_c)
          ELSE
            ratio=1.0
          END IF
          avvol_wca3a=avvolws(iwca3a,iconvey_c)-add_ws_s151
          IF (chdep(n).gt.chdep(idncl)) THEN
            s9cap=discoef(n,ind1,1,1)*(chdep(n)-chdep(idncl))**pwr(n,
     &       ind1,1,1)
          ELSE
            s9cap=0.0
          END IF
          IF (iprorte(iwca3a).eq.1) THEN
C
```

```
C   DETERMINE WATER SUPPLY RELEASE THRU S-9
C
            wsdel=dqu(ic11w)*ratio
            cout=amin1(wsdel,s9cap)*ratflw
          ELSE
            cout=amin1(dqu(ic11w),s9cap,avvol_wca3a)*ratflw
          END IF
C
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            avvolws(iwca3a,iconvey_c)=avvolws(iwca3a,iconvey_c)-cout
            outflow_from_3a(iconvey_c)=outflow_from_3a(iconvey_c)+cout
            outflow_from_3a_temp(iconvey_c)=outflow_from_3a(iconvey_c)
            dqu(idncl)=dqu(idncl)-cout
          ELSE
            outflow_from_3a_temp(iconvey_c)=outflow_from_3a(iconvey_c)+
     &       cout
          END IF
C
C-----------------------------------------------------
C   PROP PUMP DIVERTING C11W OUTFLOW TO SOUTH
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(42)) THEN
          avvolq=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvolq=max(avvolq,0.0)/86400.
          CALL match (name_res_spec_code(16), 1, 6, resname, ntotres, 6,
     &      imatch)
          ilkbelt_res=imatch
          CALL match (name_res_spec_code(17), 1, 6, resname, ntotres, 6,
     &      imatch)
          ic11res=imatch
          IF (nint(volwtr(ic11res)).eq.0.or.nint(kflo(iflc11r)).gt.0)
     &     THEN
            runoff_div_south=max(avvolq-kflo(iflc11r),0.0)
          ELSE
            runoff_div_south=0.0
          END IF
          capac=amin1(runoff_div_south,discoef(n,ind1,1,1))
          cout=amin1(capac,volwtr(ilkbelt_res))
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
          END IF
C-----------------------------------------------------
C   PROP PUMP DIVERTING WESTERN C9 OUTFLOW TO SOUTH
```

```
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(43)) THEN
          avvolq=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvolq=max(avvolq,0.0)/86400.
          runoff_div_south=max(avvolq-kflo(iflc9r),0.0)
          capac=amin1(runoff_div_south,discoef(n,ind1,1,1))
          cout=capac
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
          END IF
C-----------------------------------------------------
C         S9A : SEEPAGE PUMP TO CA3
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(86)) THEN
C
          IF (nint(outf_s9) .eq. 0) THEN
            cout=max(capacfl(imgt,1)-seep_for_ws_c11w,0.0)
          ELSE
            cout=0.0
          ENDIF
          outf_s9a = cout
c         if (conv.eq.1.)
c    &    print *,hw,rmnstgs(n,ind1,iseas),avvolq,cout,capacfl(imgt,1)
c    &      ,qu(n)/86400.,'   S9AVOL ',cgseep(n,rmnstgs(n,ind1,iseas),0,
c    &      0.)/86400.,tovflc(n,rmnstgs(n,ind1,iseas),0,0.,iter,1)
c    &      /86400.
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
          END IF
C-----------------------------------------------------
C        PUMP TO C9RES FROM US27S
C-----------------------------------------------------
```

```
        ELSE IF (flnm(iflo).eq.strname_spec(87)) THEN
C
          avvolq=outf_to_c9res_via_us27s
          avvolq=max(avvolq,0.0)

          rem_capac=discoef(n,ind1,1,1)
          rem_avvol=volwtr(irsrcvr(n,ind1,1))
          cout=min(avvolq,rem_avvol,rem_capac)
          CALL match (canl_names_spec(3), 1, 5, cnm, nch, 5, ic9)
          DO kk=1,nstrctr(ic9)
            struc_name_c9(kk)=tmpname(ic9,kk)
          END DO
          CALL match (strname_spec(84), 1, 6, struc_name_c9,
     &     nstrctr(ic9), 6, istr_idxc9)
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            volwtr(irsrcvr(n,ind1,1))=volwtr(irsrcvr(n,ind1,1))-cout
            IF (istr_idxc9.ne.0) THEN
              IF (day1) discoef1=discoef(ic9,istr_idxc9,1,1)
              discoef(ic9,istr_idxc9,1,1)=max(discoef1-cout,0.0)
            END IF
          END IF
C-------------------------------------------------------
C        PUMP TO C11RES FROM C11W
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(83)) THEN
C

          DO kk=1,nstrctr(n)
            struc_name_s13a(kk)=tmpname(n,kk)
          END DO
          CALL match ('S13A  ', 1, 6, struc_name_s13a, nstrctr(n), 6,
     &     istr_s13a)
          IF (hw.gt.chdep(ic11)+hdc(ic11,iseas)) THEN
            capac_s13a=discoef(n,istr_s13a,1,1)*(hw-chdep(ic11)-
     &       hdc(ic11,iseas))**pwr(n,istr_s13a,1,1)
          ELSE
            capac_s13a=0.0
          END IF
          rem_capac=amax1(capacfl(imgt,1)-outf_to_c9res_via_us27s-
     &     amin1(capac_s13a,qu(n)/86400.,dqu_orig(n)),0.0)
          rem_pump_cap=discoef(n,ind1,1,1)-outf_to_c9res_via_us27s
          rem_avvol=volwtr(irsrcvr(n,ind1,1))
          cout=amin1(rem_avvol,rem_capac,rem_pump_cap)
```

```
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            volwtr(irsrcvr(n,ind1,1))=volwtr(irsrcvr(n,ind1,1))-cout
          END IF
C-------------------------------------------------
C        C11W DIVIDE STRUCTURE
C-------------------------------------------------
        ELSE IF (flnm(iflo).eq.'S381E ') THEN
          DO kk=1,nstrctr(ic11w)
            struc_name_s13a(kk)=tmpname(ic11w,kk)
          END DO
          CALL match ('S13A  ', 1, 6, struc_name_s13a, nstrctr(ic11w),
     &     6, istr_s13a)
          IF (chdep(ic11w).gt.chdep(ic11)+hdc(ic11,iseas)) THEN
            capac_s13a=discoef(ic11w,istr_s13a,1,1)*(chdep(ic11w)-
     &       chdep(ic11)-hdc(ic11,iseas))**pwr(ic11w,istr_s13a,1,1)
          ELSE
            capac_s13a=0.0
          END IF
          rem_capac_s13a=amax1(capac_s13a-outf_from_c11res_to_c11w,0.0)
          cout=amin1(dqu(idncl),capacws(imgt),rem_capac_s13a)
          seep_for_ws_c11w=cout
          IF (conv.eq.1.) THEN
            dqu(idncl)=dqu(idncl)-cout
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            icnl_dns=idncl
   20       icnl_dns=iup_canal_ws(icnl_dns)
            IF (icnl_dns.le.nch) THEN
              dqu(icnl_dns)=dqu(icnl_dns)-cout
              dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
              GO TO 20
            END IF
          END IF
C-------------------------------------------------------
C            S-9
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(85)) THEN
C
          cout=amin1(outflow_from_us27s+outflow_from_us27n,capacfl(imgt,
     &     1))
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
```

```
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
          END IF
C-------------------------------------------------------
C         OUTFLOW FROM C-4 TO C-2
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(44)) THEN
C
          avvolq=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvolq=max(avvolq,0.0)/86400.
C
          avvoldn=(chdep(n)-hdc(idncl,iseas)-chdep(idncl))*cl(idncl)*
     &     width(idncl)-cgseep(idncl,chdep(n)-hdc(idncl,iseas),0,0.)-
     &     tovflc(idncl,chdep(n)-hdc(idncl,iseas),0,0.,iter,1)-qinfldn+
     &     coutlim*86400.
          avvoldn=max(avvoldn,0.0)/86400.
C
          cout_to_c2=max(avvolq,s336_to_biscbay)
          cout=min(capacfl(imgt,1),cout_to_c2,avvoldn)
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
          END IF
C-----------------------------------------------------------------
C              S-12
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(45)) THEN
C
          iconvey_c=iconvey_c_s151
          iup_wca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s12cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s12cap=0.0
          END IF
```

```
          s12_capac=s12cap
C
          IF (irceive(n,ind1).eq.1) THEN
            npts2=0
            cstor=0.0
            DO in=1,idncl-1
              npts2=npts2+nodcr(in)
            END DO
C
            rfin(idncl)=0.0
            etout(idncl)=0.0
            dsum=0.0
            voldn=0.0
            mean_land_surf_along_canal=0.0
            DO ic=1,nodcr(idncl)
              npts2=npts2+1
              nodec(ic)=nodecn(npts2)/10
              ilgth(ic)=mod(nodecn(npts2),10)
              etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
              rf=rain(nodec(ic))/(12.)
              rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
              etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &         width(idncl)
              dsum=dsum+distnc(ilgth(ic))
              hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
              rcar=distnc(ilgth(ic))*width(idncl)
              gw_depth=(ells(nodec(ic))-h(nodec(ic)))*s(nodec(ic))
              IF (pond(nodec(ic)).gt.gw_depth) THEN
                tw_node_stg=pond(nodec(ic))-gw_depth+ells(nodec(ic))
              ELSE
                tw_node_stg=(pond(nodec(ic))-gw_depth)/s(nodec(ic))+
     &           ells(nodec(ic))
              END IF
              swl_max=0.75*chdep(n)+0.25*tw_node_stg
              cstor=(swl_max-chdep(idncl))*rcar+cstor
              gwstor=max(min(ells(nodec(ic)),swl_max)-tw_node_stg,
     &         0.0)*s(nodec(ic))
              sw_incr_max=swl_max-amax1(ells(nodec(ic)),tw_node_stg)
              sw_incr_max=amax1(sw_incr_max,0.0)
              total_depth_max=amin1(1.17,gwstor+sw_incr_max)
              voldn=total_depth_max*gdar+voldn
              mean_land_surf_along_canal=mean_land_surf_along_canal+
     &         ells(nodec(ic))/nodcr(idncl)
            END DO
            swl_land_surf=mean_land_surf_along_canal-hdc(idncl,iseas)/
     &       2.0
            avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
     &       width(idncl)
            avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)-
     &       abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-rfin(idncl)+
     &       etout(idncl)
            IF (swl_max.gt.swl_land_surf) THEN
              IF (tovflc(idncl,swl_max,0,0.,iter,2).lt.0.0) THEN
                ovlnf_diff=abs(tovflc(idncl,swl_max,0,0.,iter,2))-
     &           abs(amin1(tovflc(idncl,swl_land_surf,0,0.,iter,2),0.0))
                ovlnf_diff=amax1(ovlnf_diff,0.0)
                IF (nint(ovlnf_diff).gt.0) THEN
```

```
            ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
          ELSE
            ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
          END IF
        ELSE
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
      ELSE
        ovlnf_diff=0.0
        ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
      END IF
      cstor=cstor-cgseep(idncl,swl_max,0,0.)-abs(ovlnf_int)-
&      rfin(idncl)+etout(idncl)+ovlnf_diff
      cstor=max(cstor,0.0)/86400.
      avail_cstor=max(avail_cstor,0.0)/86400.
      IF (cstor.lt.avail_cstor) THEN
        avvol_dn=cstor
      ELSE
        avvol_dn=avail_cstor+max(voldn/86400.,0.0)
      END IF
    ELSE
      avvol_dn=0.0
      DO i=1,ncells_rec(n,ind1)
        gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
&        s(nodrcv(n,ind1,i))
        IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
          tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
&          ells(nodrcv(n,ind1,i))
        ELSE
          tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
&          s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
        END IF
        swl_max=0.75*hw+0.25*tw_node_stg
        gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
        gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
        avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
&        max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
&        gw_dpth)*gdar
        avvol_dn=max(avvol,0.0)/86400.+avvol_dn
      END DO
    END IF
    IF (type_flow_across_ttrail.eq.'STAGETG') THEN
      depth_below_target=stage_import_wca(iwenp)-offset(iwenp,1)-
&      avg_sim_wca_stg(iwenp)
      depth_below_target=max(depth_below_target,0.0)
      frac_cap=min(depth_below_target/0.2,1.0)
      IF (avg_sim_wca_stg_out(ig3a28_enp)
&      .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,2))
&      THEN
        depth_above_sec_offset=avg_sim_wca_stg_out(ig3a28_enp)-
&        stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,2)
        depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
        frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
        IF (.not.lwslo(iwenp)) THEN
          frac_cap_outflow=frac_cap_up
        ELSE
```

```
                frac_cap_outflow=amax1(frac_cap_up,frac_cap)
              END IF
            ELSE IF (avg_sim_wca_stg_out(ig3a28_enp)
     &       .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,1))
     &       THEN
              depth_above_noact_zone=avg_sim_wca_stg_out(ig3a28_enp)-
     &         stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,1)
              depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
              frac_cap_up=depth_above_noact_zone/(0.25*
     &         (offset_out(ig3a28_enp,2)-offset_out(ig3a28_enp,1)))
              frac_cap_up=amin1(frac_cap_up,1.0)
              frac_cap_outflow=amin1(frac_cap,frac_cap_up)
            ELSE
              frac_cap_up=0.0
              frac_cap_outflow=0.0
            END IF
          ELSE
            frac_cap_outflow=1.0
          END IF
C
          s12cap=min(s12cap,avvol_dn)*frac_cap_outflow
C          if (conv.eq.1)
C    $      print *,s12cap,frac_cap,frac_cap_up,frac_cap_outflow,' S12'
C    $        ,LWSLO(iwenp),iwenp,stage_import_wca(iwenp)
C    $        ,avg_sim_wca_stg(iwenp),avvol_dn,s12_capac
C    $        ,rmax_depth_diff,stage_import_wca(ig3a28_enp)
C    $        ,avg_sim_wca_stg(ig3a28_enp),rmax_diff_below_targ(n,ind1)
C    $        ,rmin_diff_below_targ(n,ind1),month,iday
C          endif
C          if (conv .eq. 1) then
C          print *,s12_max_up,avvol_dn,cstor,voldn/86400.,avail_cstor
C    $                ,month,iday,s12cap,swl,MEAN_LAND_SURF_ALONG_CANAL
C    $                ,chdep(idncl),'  S-12',s12_capac
C          endif
C
          qmdel=volenp(month)/86400.
          IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
            IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,
     &       itop_zone_indx(iup_wca+1)).and.wcaenv(idn_wca_indx(is8bsn-
     &       6)).eq.'FALSE') THEN
              IF (stg3a28.gt.9.5) THEN
                qs12=2667.*(stg3a28-9.5)+1600.
              ELSE IF (stg3a28.gt.8.5) THEN
                qs12=1100.*(stg3a28-8.5)+500.
              ELSE IF (stg3a28.gt.7.5) THEN
                qs12=500.*(stg3a28-7.5)
              ELSE
                qs12=0.0
              END IF
C
              IF (type_flow_across_ttrail.eq.'STAGETG') THEN
                IF (lwslo(iwenp)) THEN
                  s12_reg=0.0
                  s12_env=amin1(qs12,s12cap)
                ELSE
                  s12_reg=amin1(qs12,s12cap)
                  s12_env=0.0
```

```
              END IF
              down_env_need=s12_env
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            ELSE
              IF (type_target.eq.'TOTAL'
     &         .or.type_flow_across_ttrail.eq.'RFPLAN ') THEN
                down_env_need=rfcfs12(iseas)*frf
              ELSE
                down_env_need=srsdmnd_w(jday)
              END IF
              s12flow=amin1(qs12,s12cap)
              s12_reg=amax1(s12flow-down_env_need,0.0)
              s12_env=amin1(down_env_need,s12flow)
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            END IF
            wca3zone='A'
C
          ELSE
            IF (type_flow_across_ttrail.ne.'STAGETG') THEN
              IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1))
     &         THEN
                IF (month.gt.10.or.month.le.5) THEN
                  total_frac=total_frac_s12(2,iseas)
                  total_frac_reg=total_frac_s12_reg(2,iseas)
                  wca3zone='C'
                ELSE
                  total_frac=total_frac_s12(1,iseas)
                  total_frac_reg=total_frac_s12_reg(1,iseas)
                  wca3zone='B'
                END IF
              ELSE IF (trig_wca_stg(iup_wca).gt.regd) THEN
                total_frac=total_frac_s12(3,iseas)
                total_frac_reg=total_frac_s12_reg(3,iseas)
                wca3zone='D'
              ELSE
                total_frac=total_frac_s12(4,iseas)
                total_frac_reg=0.0
                wca3zone='E'
              END IF
              IF (type_target.eq.'TOTAL') THEN
                rnonreg_targ_not_met_by_s333=amax1(rfcfs355(iseas)*
     &           frf-kflo(ifls355)-(kflo(ifls333)-
     &           add_s333_above_const),0.0)
                rmax_frac_targ_s12=1.0-rfcfs355(iseas)
                pot_add_nonreg_s12=(total_frac-rmax_frac_targ_s12)*
     &           frf
                add_nonreg_s12=amin1(pot_add_nonreg_s12,
     &           rnonreg_targ_not_met_by_s333)
                IF (trig_wca_stg(iup_wca).gt.regd) THEN
                  qs12t=rfcfs12(iseas)*frf+rfcfs12_reg(iseas)*(totf-
     &             frf)+rmflws333
```

```
                s12qmax=total_frac*frf+total_frac_reg*(totf-frf)
                qs12=amin1(qs12t,s12qmax)
              ELSE
                qs12t=rfcfs12(iseas)*frf+rmflws333
                qs12=amin1(qs12t,total_frac*frf)
              END IF
              IF (conv.eq.1.) THEN
                s12_flow_targ=s12_flow_targ+(qs12t-rmflws333)
              END IF
              rnon_reg_comp=rfcfs12(iseas)*frf+add_nonreg_s12
            ELSE
              qs12t=totf_w+rmflws333
              qs12=amin1(qs12t,total_frac*(totf_w+totf_e))
              qs12=amax1(qs12,totf_w)
              rnon_reg_comp=srsdmnd_w(jday)+
&              rnonreg_targ_not_met_by_s333
              IF (conv.eq.1.) THEN
                s12_flow_targ=s12_flow_targ+(qs12t-rmflws333)
              END IF
            END IF
            s12flow=amin1(qs12,s12cap)
            s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
            s12_env=amin1(s12flow,rnon_reg_comp)
            down_env_need=qs12
          ELSE
            qs12=s12cap
            IF (lwslo(iwenp)) THEN
              s12_env=qs12
              s12_reg=0.0
              down_env_need=qs12
            ELSE
              s12_env=0.0
              s12_reg=qs12
              down_env_need=0.0
            END IF
            IF (conv.eq.1.) THEN
              s12_reg_total=s12_reg+s12_reg_total
              s12_env_total=s12_env+s12_env_total
            END IF
          END IF
        END IF
      ELSE
        IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1)) THEN
          IF (stg3a28.gt.9.5) THEN
            qs12=2667.*(stg3a28-9.5)+1600.
          ELSE IF (stg3a28.gt.8.5) THEN
            qs12=1100.*(stg3a28-8.5)+500.
          ELSE IF (stg3a28.gt.7.5) THEN
            qs12=500.*(stg3a28-7.5)
          ELSE
            qs12=0.0
          END IF
          qs12=amax1(qs12,qmdel)
        ELSE
          qs12=qmdel
        END IF
        down_env_need=qmdel
```

```
          END IF
          cout=amin1(qs12,s12cap)
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            IF (type_flow_across_ttrail.eq.'MINDEL '.and.chdep(ica3)
     &       .lt.floor_elev(iwca3a,iconvey_c)) THEN
C              RATFLW = COUT / QMDEL
C              RATFLW = AMIN1(RATFLW,1.0)
              ratflw=1.0
            END IF
            rlocal_flow=0.0
            avvol_up=-901.
C
            kflo(iflo)=cout
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         chdep(idncl),s12cap,avvol_up,cout+rlocal_flow,
     &         down_env_need
            END IF
          END IF
C-----------------------------------------------------------------
C              S-12A
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(46)) THEN
C
c         CALL match (cnm(n), 1, 5, convey_canal_names, nconvey_total,
c    &     5, imatch)
c         DO isa=1,nwcas
c           IF (imatch.le.n_conv_tot(isa)) THEN
c             IF (isa.gt.1) THEN
c               iconvey_c=imatch-n_conv_tot(isa-1)
c             ELSE
c               iconvey_c=imatch
c             END IF
c             GO TO 72
c           END IF
c         END DO
          iup_wca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s12cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s12cap=0.0
          END IF
          s12_capac=s12cap
          iflood_release_override=0
          idrawdwnc=0
```

```
      DO ll=1,idrwn
        IF (n.eq.icanal_no_drawdown(ll)
     &   .and.ind1.eq.istruc_no_drawdown(ll)) THEN
         idrawdwnc=1
        END IF
      END DO
      DO i=1,4
        IF (chdep(n).gt.rmin_stg_for_outf_s12(i,month)
     &   .or.(nstorm_day(ndays_sim_sd).eq.1.and.idrawdwnc.eq.1))THEN
         iflood_release_override=1
        END IF
      END DO
      iconvey_c=iconvey_canal_no(is150bws-6)
C
C
      floor=floor_elev(iwca3a,iconvey_c)
      cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &  cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
      aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &  tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
      avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
     &  -outflow_from_3a_temp(iconvey_c)
      avvol_tot=max(avvol_tot,0.0)
      avvolws(iup_wca,iconvey_c)=avvol_tot
      IF (iprorte(iwca3a).ne.1) avvolws(iup_wca,iconvey_c)=
     &  max(avvol_tot-total_flow_s12,0.0)
C
      IF (irceive(n,ind1).eq.1) THEN
        npts2=0
        cstor=0.0
        DO in=1,idncl-1
          npts2=npts2+nodcr(in)
        END DO
C
        rfin(idncl)=0.0
        etout(idncl)=0.0
        dsum=0.0
        voldn=0.0
        mean_land_surf_along_canal=0.0
        DO ic=1,nodcr(idncl)
          npts2=npts2+1
          nodec(ic)=nodecn(npts2)/10
          ilgth(ic)=mod(nodecn(npts2),10)
          etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
          rf=rain(nodec(ic))/(12.)
          rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
          etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &     width(idncl)
          dsum=dsum+distnc(ilgth(ic))
          hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
          rcar=distnc(ilgth(ic))*width(idncl)
          gw_depth=(ells(nodec(ic))-h(nodec(ic)))*s(nodec(ic))
          IF (pond(nodec(ic)).gt.gw_depth) THEN
            tw_node_stg=pond(nodec(ic))-gw_depth+ells(nodec(ic))
          ELSE
            tw_node_stg=(pond(nodec(ic))-gw_depth)/s(nodec(ic))+
     &       ells(nodec(ic))
```

```
          END IF
          swl_max=0.75*chdep(n)+0.25*tw_node_stg
          cstor=(swl_max-chdep(idncl))*rcar+cstor
          gwstor=max(min(ells(nodec(ic)),swl_max)-tw_node_stg,
&          0.0)*s(nodec(ic))
          sw_incr_max=swl_max-amax1(ells(nodec(ic)),tw_node_stg)
          sw_incr_max=amax1(sw_incr_max,0.0)
          total_depth_max=amin1(1.17,gwstor+sw_incr_max)
          voldn=total_depth_max*gdar+voldn
          mean_land_surf_along_canal=mean_land_surf_along_canal+
&          ells(nodec(ic))/nodcr(idncl)
        END DO
        swl_land_surf=mean_land_surf_along_canal-hdc(idncl,iseas)/
&       2.0
        avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
&       width(idncl)
        avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)-
&       abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-rfin(idncl)+
&       etout(idncl)
        IF (swl_max.gt.swl_land_surf) THEN
          IF (tovflc(idncl,swl_max,0,0.,iter,2).lt.0.0) THEN
            ovlnf_diff=abs(tovflc(idncl,swl_max,0,0.,iter,2))-
&            abs(amin1(tovflc(idncl,swl_land_surf,0,0.,iter,2),0.0))
            ovlnf_diff=amax1(ovlnf_diff,0.0)
            IF (nint(ovlnf_diff).gt.0) THEN
              ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
            ELSE
              ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
            END IF
          ELSE
            ovlnf_diff=0.0
            ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
          END IF
        ELSE
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
        cstor=cstor-cgseep(idncl,swl_max,0,0.)-abs(ovlnf_int)-
&       rfin(idncl)+etout(idncl)+ovlnf_diff
        cstor=max(cstor,0.0)/86400.
        avail_cstor=max(avail_cstor,0.0)/86400.
        IF (cstor.lt.avail_cstor) THEN
          avvol_dn=cstor
        ELSE
          avvol_dn=avail_cstor+max(voldn/86400.,0.0)
        END IF
      ELSE
        avvol_dn=0.0
        DO i=1,ncells_rec(n,ind1)
          gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
&          s(nodrcv(n,ind1,i))
          IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
            tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
&            ells(nodrcv(n,ind1,i))
          ELSE
            tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
&            s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
```

```
              END IF
              swl_max=0.75*hw+0.25*tw_node_stg
              gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
              gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
              avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
     &         max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
     &         gw_dpth)*gdar
              avvol_dn=max(avvol,0.0)/86400.+avvol_dn
            END DO
          END IF
          IF (type_flow_across_ttrail.eq.'STAGETG') THEN
            depth_below_target=stage_import_wca(iwenp)-offset(iwenp,1)-
     &       avg_sim_wca_stg(iwenp)
            depth_below_target=max(depth_below_target,0.0)
            frac_cap=min(depth_below_target/0.2,1.0)
            IF (avg_sim_wca_stg_out(ig3a28_enp)
     &       .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,2))
     &       THEN
              depth_above_sec_offset=avg_sim_wca_stg_out(ig3a28_enp)-
     &         stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,2)
              depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
              frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
              IF (.not.lwslo(iwenp)) THEN
                frac_cap_outflow=frac_cap_up
              ELSE
                frac_cap_outflow=amax1(frac_cap_up,frac_cap)
              END IF
            ELSE IF (avg_sim_wca_stg_out(ig3a28_enp)
     &       .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,1))
     &       THEN
              depth_above_noact_zone=avg_sim_wca_stg_out(ig3a28_enp)-
     &         stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,1)
              depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
              frac_cap_up=depth_above_noact_zone/(0.25*
     &         (offset_out(ig3a28_enp,2)-offset_out(ig3a28_enp,1)))
              frac_cap_up=amin1(frac_cap_up,1.0)
              frac_cap_outflow=amin1(frac_cap,frac_cap_up)
            ELSE
              frac_cap_up=0.0
              frac_cap_outflow=0.0
            END IF
          ELSE
            frac_cap_outflow=1.0
          END IF
C
          s12cap=min(s12cap,avvol_dn)*frac_cap_outflow
          tw_for_constrnt=0.0
          DO k=1,no_cells_trig_s12s343(2)
            tw_for_constrnt=h(icell_loc_trig_s12s343(2))+
     &       pond(icell_loc_trig_s12s343(2))+tw_for_constrnt
          END DO
          tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(2)
          IF (tw_for_constrnt.gt.rmax_stg_s12s343(2,month)) THEN
            s12cap=0.0
          END IF
C          if (conv.eq.1)
C    $      print *,s12cap,frac_cap,frac_cap_up,frac_cap_outflow,' S12'
```

```
C     $        ,LWSLO(iwenp),iwenp,stage_import_wca(iwenp)
C     $        ,avg_sim_wca_stg(iwenp),avvol_dn,s12_capac
C     $        ,rmax_depth_diff,stage_import_wca(ig3a28_enp)
C     $        ,avg_sim_wca_stg(ig3a28_enp),rmax_diff_below_targ(n,ind1)
C     $        ,rmin_diff_below_targ(n,ind1),month,iday
C          endif
C           if (conv .eq. 1) then
C           print *,s12_max_up,avvol_dn,cstor,voldn/86400.,avail_cstor
C     $                ,month,iday,s12cap,swl,MEAN_LAND_SURF_ALONG_CANAL
C     $                ,chdep(idncl),'  S-12',s12_capac
C           endif
C
            qmdel=volenp(month)/86400.
            IF (month.eq.7.and.iday.gt.15) THEN
              frac_targs12_31m=ftargs12(3,1,month+1)
              frac_targs12_21m=ftargs12(2,1,month+1)
              frac_targs12_11m=ftargs12(1,1,month+1)
            ELSE
              frac_targs12_31m=ftargs12(3,1,month)
              frac_targs12_21m=ftargs12(2,1,month)
              frac_targs12_11m=ftargs12(1,1,month)
            END IF
            IF (iflood_release_override.eq.0) THEN
              IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
                IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,
     &           itop_zone_indx(iup_wca+1))
     &           .and.wcaenv(idn_wca_indx(is8bsn-6)).eq.'FALSE') THEN
                 IF (stg3a28.gt.9.5) THEN
                   qs12=2667.*(stg3a28-9.5)+1600.
                 ELSE IF (stg3a28.gt.8.5) THEN
                   qs12=1100.*(stg3a28-8.5)+500.
                 ELSE IF (stg3a28.gt.7.5) THEN
                   qs12=500.*(stg3a28-7.5)
                 ELSE
                   qs12=0.0
                 END IF
                 qs12=qs12*frac_targs12_31m
                 IF (type_flow_across_ttrail.eq.'STAGETG') THEN
                   IF (lwslo(iwenp)) THEN
                     s12_reg=0.0
                     s12_env=amin1(qs12,s12cap)
                   ELSE
                     s12_reg=amin1(qs12,s12cap)
                     s12_env=0.0
                   END IF
                   down_env_need=s12_env
                   IF (conv.eq.1.) THEN
                     s12_reg_total=s12_reg+s12_reg_total
                     s12_env_total=s12_env+s12_env_total
                   END IF
                 ELSE
                   IF (type_target.eq.'TOTAL'
     &              .or.type_flow_across_ttrail.eq.'RFPLAN ') THEN
                     down_env_need=frac_targs12_21m*rfcfs12(iseas)*frf
                   ELSE
                     down_env_need=frac_targs12_21m*srsdmnd_w(jday)
                   END IF
```

```
                s12flow=amin1(qs12,s12cap)
                s12_reg=amax1(s12flow-down_env_need,0.0)
                s12_env=amin1(down_env_need,s12flow)
                IF (conv.eq.1.) THEN
                  s12_reg_total=s12_reg+s12_reg_total
                  s12_env_total=s12_env+s12_env_total
                END IF
              END IF
              wca3zone='A'
              IF (conv.eq.1.) s12_flow_targ=min(s12_capac,qs12,
     &         avvol_dn)+s12_flow_targ
C
            ELSE
              IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1))
     &           THEN
                  IF (month.gt.10.or.month.le.5) THEN
                    total_frac=total_frac_s12(2,iseas)
                    total_frac_reg=total_frac_s12_reg(2,iseas)
                    wca3zone='C'
                  ELSE
                    total_frac=total_frac_s12(1,iseas)
                    total_frac_reg=total_frac_s12_reg(1,iseas)
                    wca3zone='B'
                  END IF
                ELSE IF (trig_wca_stg(iup_wca).gt.regd) THEN
                  total_frac=total_frac_s12(3,iseas)
                  total_frac_reg=total_frac_s12_reg(3,iseas)
                  wca3zone='D'
                ELSE
                  total_frac=total_frac_s12(4,iseas)
                  total_frac_reg=total_frac_s12_reg(4,iseas)
                  wca3zone='E'
                END IF
                IF (type_target.eq.'TOTAL') THEN
                  IF (iprorte(iwca3a).eq.1) THEN
                    IF (nint(frf).gt.0) THEN
                      ratio=(avvolws(iwca3a,iconvey_c)+
     &                 sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                    ELSE
                      ratio=1.0
                    END IF
                  ELSE
                    ratio=1.0
                  END IF
                  ratio=amin1(ratio,1.0)
                  rnonreg_targ_not_met_by_s333=amax1(rfcfs355(iseas)*
     &             frf*ratio-kflo(ifls355)-(kflo(ifls333)-
     &             add_s333_above_const),0.0)
                  rmax_frac_targ_s12=1.0-rfcfs355(iseas)
                  pot_add_nonreg_s12=(total_frac-rmax_frac_targ_s12)*
     &             frf*ratio
                  add_nonreg_s12=amin1(pot_add_nonreg_s12,
     &             rnonreg_targ_not_met_by_s333)
                  IF (trig_wca_stg(iup_wca).gt.regd) THEN
                    qs12t=rfcfs12(iseas)*frf+rfcfs12_reg(iseas)*(totf-
     &               frf)+rmflws333_temp
```

```
                    IF (conv.eq.1.) THEN
                      s12_flow_targ=qs12t-rmflws333
                    END IF
                    s12qmax=total_frac*frf+total_frac_reg*(totf-frf)
                    qs12t=amin1(qs12t,s12qmax)
                    qs12=frac_targs12_21m*qs12t+rmflws12
                    rnon_reg_comp=frac_targs12_21m*(rfcfs12(iseas)*
     &               frf+add_nonreg_s12)
                  ELSE
                    IF (frf.lt.250.0) THEN
                      ftargs12_low=frac_targs12_11m
                    ELSE
                      ftargs12_low=frac_targs12_21m
                    END IF
C
                    IF (iprorte(iwca3a).eq.1) THEN
C
                      qs12t=rfcfs12(iseas)*frf*ratio+rmflws333_temp
                      rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf*
     &                 ratio+add_nonreg_s12)
                    ELSE
                      qs12t=rfcfs12(iseas)*frf+rmflws333_temp
                      qs12t=amin1(qs12t,total_frac*frf,avvolws(iwca3a,
     &                 iconvey_c))
                      rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf+
     &                 add_nonreg_s12)
                      rnon_reg_comp=amin1(rnon_reg_comp,
     &                 avvolws(iwca3a,iconvey_c))
                    END IF
                    qs12t=amin1(qs12t,total_frac*frf)
                    qs12=ftargs12_low*qs12t+rmflws12
                    IF (conv.eq.1.) THEN
                      s12_flow_targ=rfcfs12(iseas)*frf
                    END IF
                  END IF
                  s12flow=amin1(qs12,s12cap)
                  IF (iprorte(iwca3a).ne.1) THEN
                    s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                  END IF
                  s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
                  s12_env=amin1(s12flow,rnon_reg_comp)
                ELSE
                  IF (iprorte(iwca3a).eq.1) THEN
                    IF (nint(totf_w).gt.0) THEN
                      ratio=(avvolws(iwca3a,iconvey_c)+
     &                 sta34_out_to_wca3ane)/(rem_wsneed_s151+totf_w+
     &                 totf_e)
                    ELSE
                      ratio=1.0
                    END IF
                  ELSE
                    ratio=1.0
                  END IF
C
                  ratio=amin1(ratio,1.0)
                  rnonreg_targ_not_met_by_s333=amax1(srsdmnd_e(jday)*
     &             ratio-kflo(ifls355)-kflo(ifls333),0.0)
```

```
                  rnonreg_targ_not_met_by_s333=
     &             amin1(rnonreg_targ_not_met_by_s333,rmflws333_temp)
C
                  IF (iprorte(iwca3a).eq.1) THEN
                    qs12t=totf_w*ratio+rmflws333_temp
                    rnon_reg_comp=frac_targs12_21m*(srsdmnd_w(jday)*
     &               ratio+rnonreg_targ_not_met_by_s333)
                  ELSE
                    qs12t=totf_w+rmflws333_temp
                    qs12t=amin1(qs12t,avvolws(iwca3a,iconvey_c))
                    rnon_reg_comp=frac_targs12_21m*(srsdmnd_w(jday)+
     &               rnonreg_targ_not_met_by_s333)
                    rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iwca3a,
     &               iconvey_c))
                    ratio=1.0
                  END IF
                  qs12t=amin1(qs12t,total_frac*(totf_w+totf_e))
                  qs12t=amax1(qs12t,totf_w*ratio)
                  qs12=frac_targs12_21m*qs12t+rmflws12
                  s12flow=amin1(qs12,s12cap)
                  IF (iprorte(iwca3a).ne.1) THEN
                    s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                  END IF
                  s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
                  s12_env=amin1(s12flow,rnon_reg_comp)
                  IF (conv.eq.1.) THEN
                    s12_flow_targ=totf_w
                  END IF
                END IF
                IF (conv.eq.1.) THEN
                  s12_reg_total=s12_reg+s12_reg_total
                  s12_env_total=s12_env+s12_env_total
                END IF
                down_env_need=qs12
              ELSE
                qs12=s12cap
                IF (lwslo(iwenp)) THEN
                  s12_env=qs12
                  s12_reg=0.0
                  down_env_need=qs12
                ELSE
                  s12_env=0.0
                  s12_reg=qs12
                  down_env_need=0.0
                END IF
                IF (conv.eq.1.) THEN
                  s12_reg_total=s12_reg+s12_reg_total
                  s12_env_total=s12_env+s12_env_total
                END IF
              END IF
            END IF
            cout=amin1(qs12,s12cap)
          ELSE
            IF (stg3a28.gt.9.5) THEN
              qs12rg=2667.*(stg3a28-9.5)+1600.
            ELSE IF (stg3a28.gt.8.5) THEN
              qs12rg=1100.*(stg3a28-8.5)+500.
```

```
            ELSE IF (stg3a28.gt.7.5) THEN
              qs12rg=500.*(stg3a28-7.5)
            ELSE
              qs12rg=0.0
            END IF
            qs12rg_tot=qs12rg
            qs12rg=qs12rg*frac_targs12_21m
            IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1)) THEN
              qs12=s12cap
              cout=amin1(qs12,qs12rg)
              IF (qs12rg_tot.lt.qmdel) THEN
                qs12mindel=frac_targs12_21m*qmdel
                cout=amin1(qs12mindel,qs12)
              END IF
            ELSE
              qs12=frac_targs12_21m*(qmdel+rmflws333_temp)+rmflws12
              cout=amin1(qs12,s12cap)
            END IF
            down_env_need=frac_targs12_21m*(qmdel+rmflws333_temp)+
     &       rmflws12
          END IF
          IF (trig_wca_stg(iup_wca).lt.regstg_wca(iup_wca,
     &     itop_zone_indx(iup_wca+1)).or.wcaenv(idn_wca_indx(is8bsn-6)
     &     ).eq.'TRUE ') THEN
            IF (frac_targs12_21m.lt.0.0001) THEN
              cout=0.0
            END IF
          END IF
          rmflws12a=qs12-cout
        ELSE
          cout=min(s12_capac,avvol_dn)
          IF (conv.eq.1.) s12_flow_targ=cout+s12_flow_targ
          rmflws12a=0.0
        END IF
C
        IF (.not.lflwtrg(ids_trg_idx(n,ind1))) THEN
          CALL match (flnm(iflo), 1, 6, struc_name_w_targ,
     &      no_of_struc_w_ds_targ_no_res, 6, istruc_targ_indx)
          cout=cout*frc_flow_targ(istruc_targ_indx)
        END IF
C
        total_flow_s12=total_flow_s12+cout
        IF (conv.eq.1) THEN
          IF (irceive(n,ind1).eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
          ELSE
            DO i=1,ncells_rec(n,ind1)
              pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &         86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &         (nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
            END DO
          END IF
          rlocal_flow=0.0
          avvol_up=-901.
C
          kflo(iflo)=cout
```

```
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s12cap,avvol_up,cout+rlocal_flow,down_env_need
            END IF
          END IF
C-----------------------------------------------------------------
C              S-12B
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(47)) THEN
C
          iup_wca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s12cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s12cap=0.0
          END IF
          s12_capac=s12cap
          iflood_release_override=0
          idrawdwnc=0
          DO ll=1,idrwn
            IF (n.eq.icanal_no_drawdown(ll)
     &       .and.ind1.eq.istruc_no_drawdown(ll)) THEN
              idrawdwnc=1
            END IF
          END DO
          DO i=1,4
            IF (chdep(n).gt.rmin_stg_for_outf_s12(i,month)
     &       .or.(nstorm_day(ndays_sim_sd).eq.1.and.idrawdwnc.eq.1))THEN
              iflood_release_override=1
            END IF
          END DO
          iconvey_c=iconvey_canal_no(is150bws-6)
C
          floor=floor_elev(iwca3a,iconvey_c)
          cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &     cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
          aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &     tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
          avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
     &     -outflow_from_3a_temp(iconvey_c)
          avvol_tot=max(avvol_tot,0.0)
          avvolws(iup_wca,iconvey_c)=avvol_tot
          IF (iprorte(iwca3a).ne.1) avvolws(iup_wca,iconvey_c)=
     &     max(avvol_tot-total_flow_s12,0.0)
C
          IF (irceive(n,ind1).eq.1) THEN
            npts2=0
            cstor=0.0
            DO in=1,idncl-1
              npts2=npts2+nodcr(in)
            END DO
C
            rfin(idncl)=0.0
            etout(idncl)=0.0
            dsum=0.0
```

```
        voldn=0.0
        mean_land_surf_along_canal=0.0
        DO ic=1,nodcr(idncl)
          npts2=npts2+1
          nodec(ic)=nodecn(npts2)/10
          ilgth(ic)=mod(nodecn(npts2),10)
          etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
          rf=rain(nodec(ic))/(12.)
          rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
          etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
&          width(idncl)
          dsum=dsum+distnc(ilgth(ic))
          hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
          rcar=distnc(ilgth(ic))*width(idncl)
          gw_depth=(ells(nodec(ic))-h(nodec(ic)))*s(nodec(ic))
          IF (pond(nodec(ic)).gt.gw_depth) THEN
            tw_node_stg=pond(nodec(ic))-gw_depth+ells(nodec(ic))
          ELSE
            tw_node_stg=(pond(nodec(ic))-gw_depth)/s(nodec(ic))+
&            ells(nodec(ic))
          END IF
          swl_max=0.75*chdep(n)+0.25*tw_node_stg
          cstor=(swl_max-chdep(idncl))*rcar+cstor
          gwstor=max(min(ells(nodec(ic)),swl_max)-tw_node_stg,
&          0.0)*s(nodec(ic))
          sw_incr_max=swl_max-amax1(ells(nodec(ic)),tw_node_stg)
          sw_incr_max=amax1(sw_incr_max,0.0)
          total_depth_max=amin1(1.17,gwstor+sw_incr_max)
          voldn=total_depth_max*gdar+voldn
          mean_land_surf_along_canal=mean_land_surf_along_canal+
&          ells(nodec(ic))/nodcr(idncl)
        END DO
        swl_land_surf=mean_land_surf_along_canal-hdc(idncl,iseas)/
&        2.0
        avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
&        width(idncl)
        avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)-
&        abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-rfin(idncl)+
&        etout(idncl)
        IF (swl_max.gt.swl_land_surf) THEN
          IF (tovflc(idncl,swl_max,0,0.,iter,2).lt.0.0) THEN
            ovlnf_diff=abs(tovflc(idncl,swl_max,0,0.,iter,2))-
&            abs(amin1(tovflc(idncl,swl_land_surf,0,0.,iter,2),0.0))
            ovlnf_diff=amax1(ovlnf_diff,0.0)
            IF (nint(ovlnf_diff).gt.0) THEN
              ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
            ELSE
              ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
            END IF
          ELSE
            ovlnf_diff=0.0
            ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
          END IF
        ELSE
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
```

```
        cstor=cstor-cgseep(idncl,swl_max,0,0.)-abs(ovlnf_int)-
&        rfin(idncl)+etout(idncl)+ovlnf_diff
        cstor=max(cstor,0.0)/86400.
        avail_cstor=max(avail_cstor,0.0)/86400.
        IF (cstor.lt.avail_cstor) THEN
          avvol_dn=cstor
        ELSE
          avvol_dn=avail_cstor+max(voldn/86400.,0.0)
        END IF
      ELSE
        avvol_dn=0.0
        DO i=1,ncells_rec(n,ind1)
          gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
&          s(nodrcv(n,ind1,i))
          IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
            tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
&            ells(nodrcv(n,ind1,i))
          ELSE
            tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
&            s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
          END IF
          swl_max=0.75*hw+0.25*tw_node_stg
          gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
          gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
          avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
&          max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
&          gw_dpth)*gdar
          avvol_dn=max(avvol,0.0)/86400.+avvol_dn
        END DO
      END IF
      IF (type_flow_across_ttrail.eq.'STAGETG') THEN
        depth_below_target=stage_import_wca(iwenp)-offset(iwenp,1)-
&        avg_sim_wca_stg(iwenp)
        depth_below_target=max(depth_below_target,0.0)
        frac_cap=min(depth_below_target/0.2,1.0)
        IF (avg_sim_wca_stg_out(ig3a28_enp)
&        .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,2))
&        THEN
          depth_above_sec_offset=avg_sim_wca_stg_out(ig3a28_enp)-
&          stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,2)
          depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
          frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
          IF (.not.lwslo(iwenp)) THEN
            frac_cap_outflow=frac_cap_up
          ELSE
            frac_cap_outflow=amax1(frac_cap_up,frac_cap)
          END IF
        ELSE IF (avg_sim_wca_stg_out(ig3a28_enp)
&        .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,1))
&        THEN
          depth_above_noact_zone=avg_sim_wca_stg_out(ig3a28_enp)-
&          stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,1)
          depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
          frac_cap_up=depth_above_noact_zone/(0.25*
&          (offset_out(ig3a28_enp,2)-offset_out(ig3a28_enp,1)))
          frac_cap_up=amin1(frac_cap_up,1.0)
          frac_cap_outflow=amin1(frac_cap,frac_cap_up)
```

```
            ELSE
              frac_cap_up=0.0
              frac_cap_outflow=0.0
            END IF
          ELSE
            frac_cap_outflow=1.0
          END IF
C
          s12cap=min(s12cap,avvol_dn)*frac_cap_outflow
          tw_for_constrnt=0.0
          DO k=1,no_cells_trig_s12s343(3)
            tw_for_constrnt=h(icell_loc_trig_s12s343(3))+
     &       pond(icell_loc_trig_s12s343(3))+tw_for_constrnt
          END DO
          tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(3)
          IF (tw_for_constrnt.gt.rmax_stg_s12s343(3,month)) THEN
            s12cap=0.0
          END IF
C          if (conv.eq.1)
C    $      print *,s12cap,frac_cap,frac_cap_up,frac_cap_outflow,' S12'
C    $        ,LWSLO(iwenp),iwenp,stage_import_wca(iwenp)
C    $        ,avg_sim_wca_stg(iwenp),avvol_dn,s12_capac
C    $        ,rmax_depth_diff,stage_import_wca(ig3a28_enp)
C    $        ,avg_sim_wca_stg(ig3a28_enp),rmax_diff_below_targ(n,ind1)
C    $        ,rmin_diff_below_targ(n,ind1),month,iday
C          endif
C           if (conv .eq. 1) then
C           print *,s12_max_up,avvol_dn,cstor,voldn/86400.,avail_cstor
C    $                ,month,iday,s12cap,swl,MEAN_LAND_SURF_ALONG_CANAL
C    $                ,chdep(idncl),'  S-12',s12_capac
C           endif
C
          qmdel=volenp(month)/86400.
          IF (month.eq.7.and.iday.gt.15) THEN
            frac_targs12_32m=ftargs12(3,2,month+1)
            frac_targs12_22m=ftargs12(2,2,month+1)
            frac_targs12_12m=ftargs12(1,2,month+1)
          ELSE
            frac_targs12_32m=ftargs12(3,2,month)
            frac_targs12_22m=ftargs12(2,2,month)
            frac_targs12_12m=ftargs12(1,2,month)
          END IF
          IF (iflood_release_override.eq.0) THEN
            IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
              IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,
     &         itop_zone_indx(iup_wca+1))
     &         .and.wcaenv(idn_wca_indx(is8bsn-6)).eq.'FALSE') THEN
               IF (stg3a28.gt.9.5) THEN
                 qs12=2667.*(stg3a28-9.5)+1600.
               ELSE IF (stg3a28.gt.8.5) THEN
                 qs12=1100.*(stg3a28-8.5)+500.
               ELSE IF (stg3a28.gt.7.5) THEN
                 qs12=500.*(stg3a28-7.5)
               ELSE
                 qs12=0.0
               END IF
               qs12=qs12*frac_targs12_32m
```

```
C
                IF (type_flow_across_ttrail.eq.'STAGETG') THEN
                  IF (lwslo(iwenp)) THEN
                    s12_reg=0.0
                    s12_env=amin1(qs12,s12cap)
                  ELSE
                    s12_reg=amin1(qs12,s12cap)
                    s12_env=0.0
                  END IF
                  down_env_need=s12_env
                  IF (conv.eq.1.) THEN
                    s12_reg_total=s12_reg+s12_reg_total
                    s12_env_total=s12_env+s12_env_total
                  END IF
                ELSE
                  IF (type_target.eq.'TOTAL'
     &             .or.type_flow_across_ttrail.eq.'RFPLAN ') THEN
                    down_env_need=frac_targs12_22m*rfcfs12(iseas)*frf
                  ELSE
                    down_env_need=frac_targs12_22m*srsdmnd_w(jday)
                  END IF
                  s12flow=amin1(qs12,s12cap)
                  s12_reg=amax1(s12flow-down_env_need,0.0)
                  s12_env=amin1(down_env_need,s12flow)
                  IF (conv.eq.1.) THEN
                    s12_reg_total=s12_reg+s12_reg_total
                    s12_env_total=s12_env+s12_env_total
                  END IF
                END IF
                IF (conv.eq.1.) THEN
                  s12_flow_targ=min(s12_capac,qs12,avvol_dn)+
     &             s12_flow_targ
                END IF
C
              ELSE
                IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                  IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1))
     &             THEN
                    IF (month.gt.10.or.month.le.5) THEN
                      total_frac=total_frac_s12(2,iseas)
                      total_frac_reg=total_frac_s12_reg(2,iseas)
                    ELSE
                      total_frac=total_frac_s12(1,iseas)
                      total_frac_reg=total_frac_s12_reg(1,iseas)
                    END IF
                  ELSE IF (trig_wca_stg(iup_wca).gt.regd) THEN
                    total_frac=total_frac_s12(3,iseas)
                    total_frac_reg=total_frac_s12_reg(3,iseas)
                  ELSE
                    total_frac=total_frac_s12(4,iseas)
                    total_frac_reg=total_frac_s12_reg(4,iseas)
                  END IF
                  IF (type_target.eq.'TOTAL') THEN
                    IF (iprorte(iwca3a).eq.1) THEN
                      IF (nint(frf).gt.0) THEN
                        ratio=(avvolws(iwca3a,iconvey_c)+
     &                   sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
```

```
                    ELSE
                      ratio=1.0
                    END IF
                  ELSE
                    ratio=1.0
                  END IF
                  ratio=amin1(ratio,1.0)
                  rnonreg_targ_not_met_by_s333=amax1(rfcfs355(iseas)*
     &             frf*ratio-kflo(ifls355)-(kflo(ifls333)-
     &             add_s333_above_const),0.0)
                  rmax_frac_targ_s12=1.0-rfcfs355(iseas)
                  pot_add_nonreg_s12=(total_frac-rmax_frac_targ_s12)*
     &             frf*ratio
                  add_nonreg_s12=amin1(pot_add_nonreg_s12,
     &             rnonreg_targ_not_met_by_s333)
                  targ_not_met_s12a=amax1(frac_targs12_21m*
     &             (rfcfs12(iseas)*frf*ratio+
     &             rnonreg_targ_not_met_by_s333)-total_flow_s12,0.0)
                  IF (trig_wca_stg(iup_wca).gt.regd) THEN
                    qs12t=rfcfs12(iseas)*frf+rfcfs12_reg(iseas)*(totf-
     &               frf)+rmflws333_temp
                    IF (conv.eq.1.) THEN
                      s12_flow_targ=qs12t-rmflws333
                    END IF
                    s12qmax=total_frac*frf+total_frac_reg*(totf-frf)
                    qs12t=amin1(qs12t,s12qmax)
                    qs12=frac_targs12_22m*qs12t+rmflws12a
                    rnon_reg_comp=frac_targs12_22m*(rfcfs12(iseas)*
     &               frf+add_nonreg_s12)+amin1(rmflws12a,
     &               targ_not_met_s12a)
                  ELSE
                    IF (frf.lt.250.0) THEN
                      ftargs12_low=frac_targs12_12m
                    ELSE
                      ftargs12_low=frac_targs12_22m
                    END IF
C
                    IF (iprorte(iwca3a).eq.1) THEN
C
                      qs12t=rfcfs12(iseas)*frf*ratio+rmflws333_temp
                      rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf*
     &                 ratio+add_nonreg_s12)+amin1(rmflws12a,
     &                 targ_not_met_s12a)
                    ELSE
                      qs12t=rfcfs12(iseas)*frf+rmflws333_temp
                      qs12t=amin1(qs12t,total_frac*frf,avvolws(iwca3a,
     &                 iconvey_c))
                      rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf+
     &                 add_nonreg_s12)+amin1(rmflws12a,
     &                 targ_not_met_s12a)
                      rnon_reg_comp=amin1(rnon_reg_comp,
     &                 avvolws(iwca3a,iconvey_c))
                    END IF
                    qs12t=amin1(qs12t,total_frac*frf)
                    qs12=ftargs12_low*qs12t+rmflws12a
                    IF (conv.eq.1.) THEN
                      s12_flow_targ=rfcfs12(iseas)*frf
```

```
                    END IF
                  END IF
                  s12flow=amin1(qs12,s12cap)
                  IF (iprorte(iwca3a).ne.1) THEN
                    s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                  END IF
                  s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
                  s12_env=amin1(s12flow,rnon_reg_comp)
                ELSE
                  IF (iprorte(iwca3a).eq.1) THEN
                    IF (nint(totf_w).gt.0) THEN
                      ratio=(avvolws(iwca3a,iconvey_c)+
     &                 sta34_out_to_wca3ane)/(rem_wsneed_s151+totf_w+
     &                 totf_e)
                    ELSE
                      ratio=1.0
                    END IF
                  ELSE
                    ratio=1.0
                  END IF
C
                  ratio=amin1(ratio,1.0)
                  rnonreg_targ_not_met_by_s333=amax1(srsdmnd_e(jday)*
     &             ratio-kflo(ifls355)-kflo(ifls333),0.0)
                  rnonreg_targ_not_met_by_s333=
     &             amin1(rnonreg_targ_not_met_by_s333,rmflws333_temp)
C
                  IF (iprorte(iwca3a).eq.1) THEN
                    qs12t=totf_w*ratio+rmflws333_temp
                    rnon_reg_comp=frac_targs12_22m*(srsdmnd_w(jday)*
     &               ratio+rnonreg_targ_not_met_by_s333)
                  ELSE
                    qs12t=totf_w+rmflws333_temp
                    qs12t=amin1(qs12t,avvolws(iwca3a,iconvey_c))
                    rnon_reg_comp=frac_targs12_22m*(srsdmnd_w(jday)+
     &               rnonreg_targ_not_met_by_s333)
                    rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iwca3a,
     &               iconvey_c))
                  END IF
                  qs12t=amin1(qs12t,total_frac*(totf_w+totf_e))
                  qs12t=amax1(qs12t,totf_w*ratio)
                  qs12=frac_targs12_22m*qs12t+rmflws12a
                  s12flow=amin1(qs12,s12cap)
                  IF (iprorte(iwca3a).ne.1) THEN
                    s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                  END IF
                  targ_not_met_s12a=frac_targs12_21m*(srsdmnd_w(jday)*
     &             ratio+rnonreg_targ_not_met_by_s333)-total_flow_s12
                  targ_not_met_s12a=amax1(targ_not_met_s12a,0.0)
                  rnon_reg_comp_1=rnon_reg_comp+amin1(rmflws12a,
     &             targ_not_met_s12a)
                  s12_reg=amax1(s12flow-rnon_reg_comp_1,0.0)
                  s12_env=amin1(s12flow,rnon_reg_comp_1)
                  IF (conv.eq.1.) THEN
                    s12_flow_targ=totf_w
                  END IF
                END IF
```

```
              down_env_need=qs12
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            ELSE
              qs12=s12cap
              IF (lwslo(iwenp)) THEN
                s12_env=qs12
                s12_reg=0.0
                down_env_need=qs12
              ELSE
                s12_env=0.0
                s12_reg=qs12
                down_env_need=0.0
              END IF
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            END IF
          END IF
          cout=amin1(qs12,s12cap)
        ELSE
          IF (stg3a28.gt.9.5) THEN
            qs12rg=2667.*(stg3a28-9.5)+1600.
          ELSE IF (stg3a28.gt.8.5) THEN
            qs12rg=1100.*(stg3a28-8.5)+500.
          ELSE IF (stg3a28.gt.7.5) THEN
            qs12rg=500.*(stg3a28-7.5)
          ELSE
            qs12rg=0.0
          END IF
          qs12rg_tot=qs12rg
          qs12rg=qs12rg*frac_targs12_22m
          IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1)) THEN
            qs12=s12cap
            cout=amin1(qs12,qs12rg)
            IF (qs12rg_tot.lt.qmdel) THEN
              qs12mindel=frac_targs12_22m*qmdel
              cout=amin1(qs12mindel,qs12)
            END IF
          ELSE
            qs12=frac_targs12_22m*(qmdel+rmflws333_temp)+rmflws12a
            cout=amin1(qs12,s12cap)
          END IF
          down_env_need=frac_targs12_22m*(qmdel+rmflws333_temp)+
&          rmflws12a
        END IF
        IF (trig_wca_stg(iup_wca).lt.regstg_wca(iup_wca,
&        itop_zone_indx(iup_wca+1)).or.wcaenv(idn_wca_indx(is8bsn-6)
&        ).eq.'TRUE ') THEN
          IF (frac_targs12_22m.lt.0.0001) THEN
            cout=0.0
          END IF
        END IF
        rmflws12b=qs12-cout
```

```
          ELSE
            cout=min(s12_capac,avvol_dn)
            IF (conv.eq.1.) THEN
              s12_flow_targ=cout+s12_flow_targ
            END IF
            rmflws12b=0.0
          END IF
C
          IF (.not.lflwtrg(ids_trg_idx(n,ind1))) THEN
            CALL match (flnm(iflo), 1, 6, struc_name_w_targ,
     &       no_of_struc_w_ds_targ_no_res, 6, istruc_targ_indx)
            cout=cout*frc_flow_targ(istruc_targ_indx)
          END IF
          total_flow_s12=total_flow_s12+cout
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            rlocal_flow=0.0
            avvol_up=-901.
C
            kflo(iflo)=cout
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s12cap,avvol_up,cout+rlocal_flow,down_env_need
            END IF
          END IF
C-----------------------------------------------------------------
C              S-12C
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(48)) THEN
C
          iconvey_c=iconvey_canal_no(is150bws-6)
          iup_wca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s12cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s12cap=0.0
          END IF
          s12_capac=s12cap
          iflood_release_override=0
          idrawdwnc=0
          DO ll=1,idrwn
            IF (n.eq.icanal_no_drawdown(ll)
     &       .and.ind1.eq.istruc_no_drawdown(ll)) THEN
              idrawdwnc=1
            END IF
          END DO
```

```
      DO i=1,4
        IF (chdep(n).gt.rmin_stg_for_outf_s12(i,month)
     &   .or.(nstorm_day(ndays_sim_sd).eq.1.and.idrawdwnc.eq.1))THEN
         iflood_release_override=1
        END IF
      END DO
C
      floor=floor_elev(iwca3a,iconvey_c)
      cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &  cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
      aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &  tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
      avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
     &  -outflow_from_3a_temp(iconvey_c)
      avvol_tot=max(avvol_tot,0.0)
      avvolws(iup_wca,iconvey_c)=avvol_tot
      IF (iprorte(iwca3a).ne.1) avvolws(iup_wca,iconvey_c)=
     &  max(avvol_tot-total_flow_s12,0.0)
C
      IF (irceive(n,ind1).eq.1) THEN
        npts2=0
        cstor=0.0
        DO in=1,idncl-1
          npts2=npts2+nodcr(in)
        END DO
C
        rfin(idncl)=0.0
        etout(idncl)=0.0
        dsum=0.0
        voldn=0.0
        mean_land_surf_along_canal=0.0
        DO ic=1,nodcr(idncl)
          npts2=npts2+1
          nodec(ic)=nodecn(npts2)/10
          ilgth(ic)=mod(nodecn(npts2),10)
          etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
          rf=rain(nodec(ic))/(12.)
          rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
          etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &     width(idncl)
          dsum=dsum+distnc(ilgth(ic))
          hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
          rcar=distnc(ilgth(ic))*width(idncl)
          gw_depth=(ells(nodec(ic))-h(nodec(ic)))*s(nodec(ic))
          IF (pond(nodec(ic)).gt.gw_depth) THEN
            tw_node_stg=pond(nodec(ic))-gw_depth+ells(nodec(ic))
          ELSE
            tw_node_stg=(pond(nodec(ic))-gw_depth)/s(nodec(ic))+
     &       ells(nodec(ic))
          END IF
          swl_max=0.75*chdep(n)+0.25*tw_node_stg
          cstor=(swl_max-chdep(idncl))*rcar+cstor
          gwstor=max(min(ells(nodec(ic)),swl_max)-tw_node_stg,
     &     0.0)*s(nodec(ic))
          sw_incr_max=swl_max-amax1(ells(nodec(ic)),tw_node_stg)
          sw_incr_max=amax1(sw_incr_max,0.0)
          total_depth_max=amin1(1.17,gwstor+sw_incr_max)
```

```
          voldn=total_depth_max*gdar+voldn
          mean_land_surf_along_canal=mean_land_surf_along_canal+
&          ells(nodec(ic))/nodcr(idncl)
        END DO
        swl_land_surf=mean_land_surf_along_canal-hdc(idncl,iseas)/
&        2.0
        avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
&        width(idncl)
        avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)-
&        abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-rfin(idncl)+
&        etout(idncl)
        IF (swl_max.gt.swl_land_surf) THEN
          IF (tovflc(idncl,swl_max,0,0.,iter,2).lt.0.0) THEN
            ovlnf_diff=abs(tovflc(idncl,swl_max,0,0.,iter,2))-
&            abs(amin1(tovflc(idncl,swl_land_surf,0,0.,iter,2),0.0))
            ovlnf_diff=amax1(ovlnf_diff,0.0)
            IF (nint(ovlnf_diff).gt.0) THEN
              ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
            ELSE
              ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
            END IF
          ELSE
            ovlnf_diff=0.0
            ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
          END IF
        ELSE
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
        cstor=cstor-cgseep(idncl,swl_max,0,0.)-abs(ovlnf_int)-
&        rfin(idncl)+etout(idncl)+ovlnf_diff
        cstor=max(cstor,0.0)/86400.
        avail_cstor=max(avail_cstor,0.0)/86400.
        IF (cstor.lt.avail_cstor) THEN
          avvol_dn=cstor
        ELSE
          avvol_dn=avail_cstor+max(voldn/86400.,0.0)
        END IF
      ELSE
        avvol_dn=0.0
        DO i=1,ncells_rec(n,ind1)
          gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
&          s(nodrcv(n,ind1,i))
          IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
            tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
&            ells(nodrcv(n,ind1,i))
          ELSE
            tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
&            s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
          END IF
          swl_max=0.75*hw+0.25*tw_node_stg
          gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
          gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
          avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
&          max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
&          gw_dpth)*gdar
          avvol_dn=max(avvol,0.0)/86400.+avvol_dn
```

```
            END DO
          END IF
          IF (type_flow_across_ttrail.eq.'STAGETG') THEN
            depth_below_target=stage_import_wca(iwenp)-offset(iwenp,1)-
     &       avg_sim_wca_stg(iwenp)
            depth_below_target=max(depth_below_target,0.0)
            frac_cap=min(depth_below_target/0.2,1.0)
            IF (avg_sim_wca_stg_out(ig3a28_enp)
     &       .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,2))
     &       THEN
              depth_above_sec_offset=avg_sim_wca_stg_out(ig3a28_enp)-
     &         stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,2)
              depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
              frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
              IF (.not.lwslo(iwenp)) THEN
                frac_cap_outflow=frac_cap_up
              ELSE
                frac_cap_outflow=amax1(frac_cap_up,frac_cap)
              END IF
            ELSE IF (avg_sim_wca_stg_out(ig3a28_enp)
     &       .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,1))
     &       THEN
              depth_above_noact_zone=avg_sim_wca_stg_out(ig3a28_enp)-
     &         stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,1)
              depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
              frac_cap_up=depth_above_noact_zone/(0.25*
     &         (offset_out(ig3a28_enp,2)-offset_out(ig3a28_enp,1)))
              frac_cap_up=amin1(frac_cap_up,1.0)
              frac_cap_outflow=amin1(frac_cap,frac_cap_up)
            ELSE
              frac_cap_up=0.0
              frac_cap_outflow=0.0
            END IF
          ELSE
            frac_cap_outflow=1.0
          END IF
C
          s12cap=min(s12cap,avvol_dn)*frac_cap_outflow
          tw_for_constrnt=0.0
          DO k=1,no_cells_trig_s12s343(4)
            tw_for_constrnt=h(icell_loc_trig_s12s343(4))+
     &       pond(icell_loc_trig_s12s343(4))+tw_for_constrnt
          END DO
          tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(4)
          IF (tw_for_constrnt.gt.rmax_stg_s12s343(4,month)) THEN
            s12cap=0.0
          END IF
C          if (conv.eq.1)
C    $      print *,s12cap,frac_cap,frac_cap_up,frac_cap_outflow,' S12'
C    $        ,LWSLO(iwenp),iwenp,stage_import_wca(iwenp)
C    $        ,avg_sim_wca_stg(iwenp),avvol_dn,s12_capac
C    $        ,rmax_depth_diff,stage_import_wca(ig3a28_enp)
C    $        ,avg_sim_wca_stg(ig3a28_enp),rmax_diff_below_targ(n,ind1)
C    $        ,rmin_diff_below_targ(n,ind1),month,iday
C          endif
C           if (conv .eq. 1) then
C           print *,s12_max_up,avvol_dn,cstor,voldn/86400.,avail_cstor
```

```
C     $                  ,month,iday,s12cap,swl,MEAN_LAND_SURF_ALONG_CANAL
C     $                  ,chdep(idncl),'  S-12',s12_capac
C           endif
C
            qmdel=volenp(month)/86400.
            IF (month.eq.7.and.iday.gt.15) THEN
              frac_targs12_33m=ftargs12(3,3,month+1)
              frac_targs12_23m=ftargs12(2,3,month+1)
              frac_targs12_13m=ftargs12(1,3,month+1)
            ELSE
              frac_targs12_33m=ftargs12(3,3,month)
              frac_targs12_23m=ftargs12(2,3,month)
              frac_targs12_13m=ftargs12(1,3,month)
            END IF
            IF (iflood_release_override.eq.0) THEN
              IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
                IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,
     &           itop_zone_indx(iup_wca+1))
     &           .and.wcaenv(idn_wca_indx(is8bsn-6)).eq.'FALSE') THEN
                 IF (stg3a28.gt.9.5) THEN
                   qs12=2667.*(stg3a28-9.5)+1600.
                 ELSE IF (stg3a28.gt.8.5) THEN
                   qs12=1100.*(stg3a28-8.5)+500.
                 ELSE IF (stg3a28.gt.7.5) THEN
                   qs12=500.*(stg3a28-7.5)
                 ELSE
                   qs12=0.0
                 END IF
                 qs12=qs12*frac_targs12_33m
C
                 IF (type_flow_across_ttrail.eq.'STAGETG') THEN
                   IF (lwslo(iwenp)) THEN
                     s12_reg=0.0
                     s12_env=amin1(qs12,s12cap)
                   ELSE
                     s12_reg=amin1(qs12,s12cap)
                     s12_env=0.0
                   END IF
                   down_env_need=s12_env
                   IF (conv.eq.1) THEN
                     s12_reg_total=s12_reg+s12_reg_total
                     s12_env_total=s12_env+s12_env_total
                   END IF
                 ELSE
                   IF (type_target.eq.'TOTAL'
     &              .or.type_flow_across_ttrail.eq.'RFPLAN ') THEN
                     down_env_need=frac_targs12_23m*rfcfs12(iseas)*frf
                   ELSE
                     down_env_need=frac_targs12_23m*srsdmnd_w(jday)
                   END IF
                   s12flow=amin1(qs12,s12cap)
                   s12_reg=amax1(s12flow-down_env_need,0.0)
                   s12_env=amin1(down_env_need,s12flow)
                   IF (conv.eq.1.) THEN
                     s12_reg_total=s12_reg+s12_reg_total
                     s12_env_total=s12_env+s12_env_total
                   END IF
```

```
                END IF
                IF (conv.eq.1.) THEN
                  s12_flow_targ=min(s12_capac,qs12,avvol_dn)+
     &             s12_flow_targ
                END IF
C
              ELSE
                IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                  IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1))
     &             THEN
                    IF (month.gt.10.or.month.le.5) THEN
                      total_frac=total_frac_s12(2,iseas)
                      total_frac_reg=total_frac_s12_reg(2,iseas)
                    ELSE
                      total_frac=total_frac_s12(1,iseas)
                      total_frac_reg=total_frac_s12_reg(1,iseas)
                    END IF
                  ELSE IF (trig_wca_stg(iup_wca).gt.regd) THEN
                    total_frac=total_frac_s12(3,iseas)
                    total_frac_reg=total_frac_s12_reg(3,iseas)
                  ELSE
                    total_frac=total_frac_s12(4,iseas)
                    total_frac_reg=total_frac_s12_reg(4,iseas)
                  END IF
                  IF (type_target.eq.'TOTAL') THEN
                    IF (iprorte(iwca3a).eq.1) THEN
                      IF (nint(frf).gt.0) THEN
                        ratio=(avvolws(iwca3a,iconvey_c)+
     &                   sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                      ELSE
                        ratio=1.0
                      END IF
                    ELSE
                      ratio=1.0
                    END IF
                    ratio=amin1(ratio,1.0)
                    rnonreg_targ_not_met_by_s333=amax1(rfcfs355(iseas)*
     &               frf*ratio-kflo(ifls355)-(kflo(ifls333)-
     &               add_s333_above_const),0.0)
                    rmax_frac_targ_s12=1.0-rfcfs355(iseas)
                    pot_add_nonreg_s12=(total_frac-rmax_frac_targ_s12)*
     &               frf*ratio
                    add_nonreg_s12=amin1(pot_add_nonreg_s12,
     &               rnonreg_targ_not_met_by_s333)
                    targ_not_met_s12ab=amax1((frac_targs12_21m+
     &               frac_targs12_22m)*(rfcfs12(iseas)*frf+
     &               rnonreg_targ_not_met_by_s333)-total_flow_s12,0.0)
                    IF (trig_wca_stg(iup_wca).gt.regd) THEN
                      qs12t=rfcfs12(iseas)*frf+rfcfs12_reg(iseas)*(totf-
     &                 frf)+rmflws333_temp
                      IF (conv.eq.1.) THEN
                        s12_flow_targ=qs12t-rmflws333
                      END IF
                      s12qmax=total_frac*frf+total_frac_reg*(totf-frf)
                      qs12t=amin1(qs12t,s12qmax)
                      qs12=frac_targs12_23m*qs12t+rmflws12b
                      rnon_reg_comp=frac_targs12_23m*(rfcfs12(iseas)*
```

```
     &                 frf+add_nonreg_s12)+amin1(rmflws12b,
     &                 targ_not_met_s12ab)
                     ELSE
                       IF (frf.lt.250.0) THEN
                         ftargs12_low=frac_targs12_13m
                       ELSE
                         ftargs12_low=frac_targs12_23m
                       END IF
C
                       IF (iprorte(iwca3a).eq.1) THEN
C
                         qs12t=rfcfs12(iseas)*frf*ratio+rmflws333_temp
                         rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf*
     &                    ratio+add_nonreg_s12)+amin1(rmflws12b,
     &                    targ_not_met_s12ab)
                       ELSE
                         qs12t=rfcfs12(iseas)*frf+rmflws333_temp
                         qs12t=amin1(qs12t,total_frac*frf,avvolws(iwca3a,
     &                    iconvey_c))
                         rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf+
     &                    add_nonreg_s12)+amin1(rmflws12b,
     &                    targ_not_met_s12ab)
                         rnon_reg_comp=amin1(rnon_reg_comp,
     &                    avvolws(iwca3a,iconvey_c))
                       END IF
                       qs12t=amin1(qs12t,total_frac*frf)
                       qs12=ftargs12_low*qs12t+rmflws12b
                       IF (conv.eq.1.) THEN
                         s12_flow_targ=rfcfs12(iseas)*frf
                       END IF
                     END IF
                     s12flow=amin1(qs12,s12cap)
                     IF (iprorte(iwca3a).ne.1) THEN
                       s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                     END IF
                     s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
                     s12_env=amin1(s12flow,rnon_reg_comp)
                   ELSE
                     IF (iprorte(iwca3a).eq.1) THEN
                       IF (nint(totf_w).gt.0) THEN
                         ratio=(avvolws(iwca3a,iconvey_c)+
     &                    sta34_out_to_wca3ane)/(rem_wsneed_s151+totf_w+
     &                    totf_e)
                       ELSE
                         ratio=1.0
                       END IF
                     ELSE
                       ratio=1.0
                     END IF
C
                     ratio=amin1(ratio,1.0)
                     rnonreg_targ_not_met_by_s333=amax1(srsdmnd_e(jday)*
     &                ratio-kflo(ifls355)-kflo(ifls333),0.0)
                     rnonreg_targ_not_met_by_s333=
     &                amin1(rnonreg_targ_not_met_by_s333,rmflws333_temp)
C
                     IF (iprorte(iwca3a).eq.1) THEN
```

```
                  qs12t=totf_w*ratio+rmflws333_temp
                  rnon_reg_comp=frac_targs12_23m*(srsdmnd_w(jday)*
&                  ratio+rnonreg_targ_not_met_by_s333)
                ELSE
                  qs12t=totf_w+rmflws333_temp
                  qs12t=amin1(qs12t,avvolws(iwca3a,iconvey_c))
                  rnon_reg_comp=frac_targs12_23m*(srsdmnd_w(jday)+
&                  rnonreg_targ_not_met_by_s333)
                  rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iwca3a,
&                  iconvey_c))
                END IF
                qs12t=amin1(qs12t,total_frac*(totf_w+totf_e))
                qs12t=amax1(qs12t,totf_w)
                qs12=frac_targs12_23m*qs12t+rmflws12b
                s12flow=amin1(qs12,s12cap)
                IF (iprorte(iwca3a).ne.1) THEN
                  s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                END IF
                targ_not_met_s12ab=(frac_targs12_21m+
&                frac_targs12_22m)*(srsdmnd_w(jday)+
&                rnonreg_targ_not_met_by_s333)-total_flow_s12
                targ_not_met_s12ab=amax1(targ_not_met_s12ab,0.0)
                rnon_reg_comp_1=rnon_reg_comp+amin1(rmflws12b,
&                targ_not_met_s12ab)
                s12_reg=amax1(s12flow-rnon_reg_comp_1,0.0)
                s12_env=amin1(s12flow,rnon_reg_comp_1)
                IF (conv.eq.1.) THEN
                  s12_flow_targ=totf_w
                END IF
              END IF
              down_env_need=qs12
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            ELSE
              qs12=s12cap
              IF (lwslo(iwenp)) THEN
                s12_env=qs12
                s12_reg=0.0
                down_env_need=qs12
              ELSE
                s12_env=0.0
                s12_reg=qs12
                down_env_need=0.0
              END IF
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            END IF
          END IF
          cout=amin1(qs12,s12cap)
        ELSE
          IF (stg3a28.gt.9.5) THEN
            qs12rg=2667.*(stg3a28-9.5)+1600.
          ELSE IF (stg3a28.gt.8.5) THEN
```

```
              qs12rg=1100.*(stg3a28-8.5)+500.
            ELSE IF (stg3a28.gt.7.5) THEN
              qs12rg=500.*(stg3a28-7.5)
            ELSE
              qs12rg=0.0
            END IF
            qs12rg_tot=qs12rg
            qs12rg=qs12rg*frac_targs12_23m
            IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1)) THEN
              qs12=s12cap
              cout=amin1(qs12,qs12rg)
              IF (qs12rg_tot.lt.qmdel) THEN
                qs12mindel=frac_targs12_23m*qmdel
                cout=amin1(qs12mindel,qs12)
              END IF
            ELSE
              qs12=frac_targs12_23m*(qmdel+rmflws333_temp)+rmflws12b
              cout=amin1(qs12,s12cap)
            END IF
            down_env_need=frac_targs12_23m*(qmdel+rmflws333_temp)+
     &       rmflws12b
          END IF
          IF (trig_wca_stg(iup_wca).lt.regstg_wca(iup_wca,
     &     itop_zone_indx(iup_wca+1)).or.wcaenv(idn_wca_indx(is8bsn-6)
     &     ).eq.'TRUE ') THEN
            IF (frac_targs12_23m.lt.0.0001) THEN
              cout=0.0
            END IF
          END IF
          rmflws12c=qs12-cout
        ELSE
          cout=min(s12_capac,avvol_dn)
          IF (conv.eq.1.) THEN
            s12_flow_targ=cout+s12_flow_targ
          END IF
          rmflws12c=0.0
        END IF
C
        IF (.not.lflwtrg(ids_trg_idx(n,ind1))) THEN
          CALL match (flnm(iflo), 1, 6, struc_name_w_targ,
     &     no_of_struc_w_ds_targ_no_res, 6, istruc_targ_indx)
          cout=cout*frc_flow_targ(istruc_targ_indx)
        END IF
        total_flow_s12=total_flow_s12+cout
        IF (conv.eq.1) THEN
          IF (irceive(n,ind1).eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
          ELSE
            DO i=1,ncells_rec(n,ind1)
              pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &         86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &         (nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
            END DO
          END IF
          rlocal_flow=0.0
          avvol_up=-901.
```

```
C
            kflo(iflo)=cout
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s12cap,avvol_up,cout+rlocal_flow,down_env_need
            END IF
          END IF
C-----------------------------------------------------------------
C              S-12D
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(49)) THEN
C
          iup_wca=idn_wca_indx(is8bsn-6)
C
          iconvey_c=iconvey_canal_no(is150bws-6)
          IF (chdep(n).gt.tw) THEN
            s12cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s12cap=0.0
          END IF
          s12_capac=s12cap
C
          iflood_release_override=0
          idrawdwnc=0
          DO ll=1,idrwn
            IF (n.eq.icanal_no_drawdown(ll)
     &       .and.ind1.eq.istruc_no_drawdown(ll)) THEN
              idrawdwnc=1
            END IF
          END DO
          DO i=1,4
            IF (chdep(n).gt.rmin_stg_for_outf_s12(i,month)
     &       .or.(nstorm_day(ndays_sim_sd).eq.1.and.idrawdwnc.eq.1))THEN
              iflood_release_override=1
            END IF
          END DO
C
          floor=floor_elev(iwca3a,iconvey_c)
          cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &     cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
          aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &     tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
          avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
     &     -outflow_from_3a_temp(iconvey_c)
          avvol_tot=max(avvol_tot,0.0)
          avvolws(iup_wca,iconvey_c)=avvol_tot
          IF (iprorte(iwca3a).ne.1) avvolws(iup_wca,iconvey_c)=
     &     max(avvol_tot-total_flow_s12,0.0)
C
          IF (irceive(n,ind1).eq.1) THEN
            npts2=0
            cstor=0.0
            DO in=1,idncl-1
              npts2=npts2+nodcr(in)
            END DO
C
```

```
      rfin(idncl)=0.0
      etout(idncl)=0.0
      dsum=0.0
      voldn=0.0
      mean_land_surf_along_canal=0.0
      DO ic=1,nodcr(idncl)
        npts2=npts2+1
        nodec(ic)=nodecn(npts2)/10
        ilgth(ic)=mod(nodecn(npts2),10)
        etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
        rf=rain(nodec(ic))/(12.)
        rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
        etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
&         width(idncl)
        dsum=dsum+distnc(ilgth(ic))
        hdiff=hdc(idncl,iseas)*(cl(idncl)-dsum)/cl(idncl)
        rcar=distnc(ilgth(ic))*width(idncl)
        gw_depth=(ells(nodec(ic))-h(nodec(ic)))*s(nodec(ic))
        IF (pond(nodec(ic)).gt.gw_depth) THEN
          tw_node_stg=pond(nodec(ic))-gw_depth+ells(nodec(ic))
        ELSE
          tw_node_stg=(pond(nodec(ic))-gw_depth)/s(nodec(ic))+
&           ells(nodec(ic))
        END IF
        swl_max=0.75*chdep(n)+0.25*tw_node_stg
        cstor=(swl_max-chdep(idncl))*rcar+cstor
        gwstor=max(min(ells(nodec(ic)),swl_max)-tw_node_stg,
&         0.0)*s(nodec(ic))
        sw_incr_max=swl_max-amax1(ells(nodec(ic)),tw_node_stg)
        sw_incr_max=amax1(sw_incr_max,0.0)
        total_depth_max=amin1(2.17,gwstor+sw_incr_max)
        voldn=total_depth_max*gdar+voldn
        mean_land_surf_along_canal=mean_land_surf_along_canal+
&         ells(nodec(ic))/nodcr(idncl)
      END DO
      swl_land_surf=mean_land_surf_along_canal-hdc(idncl,iseas)/
&       2.0
      avail_cstor=(swl_land_surf-chdep(idncl))*cl(idncl)*
&       width(idncl)
      avail_cstor=avail_cstor-cgseep(idncl,swl_land_surf,0,0.)-
&       abs(tovflc(idncl,swl_land_surf,0,0.,iter,2))-rfin(idncl)+
&       etout(idncl)
      IF (swl_max.gt.swl_land_surf) THEN
        IF (tovflc(idncl,swl_max,0,0.,iter,2).lt.0.0) THEN
          ovlnf_diff=abs(tovflc(idncl,swl_max,0,0.,iter,2))-
&           abs(amin1(tovflc(idncl,swl_land_surf,0,0.,iter,2),0.0))
          ovlnf_diff=amax1(ovlnf_diff,0.0)
          IF (nint(ovlnf_diff).gt.0) THEN
            ovlnf_int=tovflc(idncl,swl_land_surf,0,0.,iter,2)
          ELSE
            ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
          END IF
        ELSE
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
      ELSE
```

```
          ovlnf_diff=0.0
          ovlnf_int=tovflc(idncl,swl_max,0,0.,iter,2)
        END IF
        cstor=cstor-cgseep(idncl,swl_max,0,0.)-abs(ovlnf_int)-
&        rfin(idncl)+etout(idncl)+ovlnf_diff
        cstor=max(cstor,0.0)/86400.
        avail_cstor=max(avail_cstor,0.0)/86400.
        IF (cstor.lt.avail_cstor) THEN
          avvol_dn=cstor
        ELSE
          avvol_dn=avail_cstor+max(voldn/86400.,0.0)
        END IF
      ELSE
        avvol_dn=0.0
        DO i=1,ncells_rec(n,ind1)
          gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
&          s(nodrcv(n,ind1,i))
          IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
            tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
&            ells(nodrcv(n,ind1,i))
          ELSE
            tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
&            s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
          END IF
          swl_max=0.75*hw+0.25*tw_node_stg
          gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
          gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
          avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
&          max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
&          gw_dpth)*gdar
          avvol_dn=max(avvol,0.0)/86400.+avvol_dn
        END DO
      END IF
      IF (type_flow_across_ttrail.eq.'STAGETG') THEN
        depth_below_target=stage_import_wca(iwenp)-offset(iwenp,1)-
&        avg_sim_wca_stg(iwenp)
        depth_below_target=max(depth_below_target,0.0)
        frac_cap=min(depth_below_target/0.2,1.0)
        IF (avg_sim_wca_stg_out(ig3a28_enp)
&        .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,2))
&        THEN
          depth_above_sec_offset=avg_sim_wca_stg_out(ig3a28_enp)-
&          stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,2)
          depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
          frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
          IF (.not.lwslo(iwenp)) THEN
            frac_cap_outflow=frac_cap_up
          ELSE
            frac_cap_outflow=amax1(frac_cap_up,frac_cap)
          END IF
        ELSE IF (avg_sim_wca_stg_out(ig3a28_enp)
&        .gt.stage_outflw_wca(ig3a28_enp)+offset_out(ig3a28_enp,1))
&        THEN
          depth_above_noact_zone=avg_sim_wca_stg_out(ig3a28_enp)-
&          stage_outflw_wca(ig3a28_enp)-offset_out(ig3a28_enp,1)
          depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
          frac_cap_up=depth_above_noact_zone/(0.25*
```

```
     &         (offset_out(ig3a28_enp,2)-offset_out(ig3a28_enp,1)))
               frac_cap_up=amin1(frac_cap_up,1.0)
               frac_cap_outflow=amin1(frac_cap,frac_cap_up)
             ELSE
               frac_cap_up=0.0
               frac_cap_outflow=0.0
             END IF
           ELSE
             frac_cap_outflow=1.0
           END IF
C
           s12cap=min(s12cap,avvol_dn)*frac_cap_outflow
           tw_for_constrnt=0.0
           DO k=1,no_cells_trig_s12s343(5)
             tw_for_constrnt=h(icell_loc_trig_s12s343(5))+
     &        pond(icell_loc_trig_s12s343(5))+tw_for_constrnt
           END DO
           tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(5)
           IF (tw_for_constrnt.gt.rmax_stg_s12s343(5,month)) THEN
             s12cap=0.0
           END IF
C           if (conv.eq.1)
C    $        print *,s12cap,frac_cap,frac_cap_up,frac_cap_outflow,' S12'
C    $          ,LWSLO(iwenp),iwenp,stage_import_wca(iwenp)
C    $          ,avg_sim_wca_stg(iwenp),avvol_dn,s12_capac
C    $          ,rmax_depth_diff,stage_import_wca(ig3a28_enp)
C    $          ,avg_sim_wca_stg(ig3a28_enp),rmax_diff_below_targ(n,ind1)
C    $          ,rmin_diff_below_targ(n,ind1),month,iday
C           endif
C            if (conv .eq. 1) then
C            print *,s12_max_up,avvol_dn,cstor,voldn/86400.,avail_cstor
C    $                  ,month,iday,s12cap,swl,MEAN_LAND_SURF_ALONG_CANAL
C    $                  ,chdep(idncl),'  S-12',s12_capac
C            endif
C
           qmdel=volenp(month)/86400.
           IF (month.eq.7.and.iday.gt.15) THEN
             frac_targs12_34m=ftargs12(3,4,month+1)
             frac_targs12_24m=ftargs12(2,4,month+1)
             frac_targs12_14m=ftargs12(1,4,month+1)
           ELSE
             frac_targs12_34m=ftargs12(3,4,month)
             frac_targs12_24m=ftargs12(2,4,month)
             frac_targs12_14m=ftargs12(1,4,month)
           END IF
           IF (iflood_release_override.eq.0) THEN
             IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
               IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,
     &          itop_zone_indx(iup_wca+1))
     &          .and.wcaenv(idn_wca_indx(is8bsn-6)).eq.'FALSE') THEN
                IF (stg3a28.gt.9.5) THEN
                  qs12=2667.*(stg3a28-9.5)+1600.
                ELSE IF (stg3a28.gt.8.5) THEN
                  qs12=1100.*(stg3a28-8.5)+500.
                ELSE IF (stg3a28.gt.7.5) THEN
                  qs12=500.*(stg3a28-7.5)
                ELSE
```

```
                  qs12=0.0
                END IF
                qs12=qs12*frac_targs12_34m
C
                IF (type_flow_across_ttrail.eq.'STAGETG') THEN
                  IF (lwslo(iwenp)) THEN
                    s12_reg=0.0
                    s12_env=amin1(qs12,s12cap)
                  ELSE
                    s12_reg=amin1(qs12,s12cap)
                    s12_env=0.0
                  END IF
                  down_env_need=s12_env
                  IF (conv.eq.1.) THEN
                    s12_reg_total=s12_reg+s12_reg_total
                    s12_env_total=s12_env+s12_env_total
                  END IF
                ELSE
                  IF (type_target.eq.'TOTAL'
     &             .or.type_flow_across_ttrail.eq.'RFPLAN ') THEN
                    down_env_need=frac_targs12_24m*rfcfs12(iseas)*frf
                  ELSE
                    down_env_need=frac_targs12_24m*srsdmnd_w(jday)
                  END IF
                  s12flow=amin1(qs12,s12cap)
                  s12_reg=amax1(s12flow-down_env_need,0.0)
                  s12_env=amin1(down_env_need,s12flow)
                  IF (conv.eq.1.) THEN
                    s12_reg_total=s12_reg+s12_reg_total
                    s12_env_total=s12_env+s12_env_total
                  END IF
                END IF
                IF (conv.eq.1.) THEN
                  s12_flow_targ=min(s12_capac,qs12,avvol_dn)+
     &             s12_flow_targ
                END IF
C
              ELSE
                IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                  IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1))
     &             THEN
                    IF (month.gt.10.or.month.le.5) THEN
                      total_frac=total_frac_s12(2,iseas)
                      total_frac_reg=total_frac_s12_reg(2,iseas)
                    ELSE
                      total_frac=total_frac_s12(1,iseas)
                      total_frac_reg=total_frac_s12_reg(1,iseas)
                    END IF
                  ELSE IF (trig_wca_stg(iup_wca).gt.regd) THEN
                    total_frac=total_frac_s12(3,iseas)
                    total_frac_reg=total_frac_s12_reg(3,iseas)
                  ELSE
                    total_frac=total_frac_s12(4,iseas)
                    total_frac_reg=total_frac_s12_reg(4,iseas)
                  END IF
                  IF (type_target.eq.'TOTAL') THEN
                    rnonreg_targ_not_met_by_s333=amax1(rfcfs355(iseas)*
```

```
     &                 frf-kflo(ifls355)-(kflo(ifls333)-
     &                 add_s333_above_const),0.0)
                      rmax_frac_targ_s12=1.0-rfcfs355(iseas)
                      pot_add_nonreg_s12=(total_frac-rmax_frac_targ_s12)*
     &                 frf
                      add_nonreg_s12=amin1(pot_add_nonreg_s12,
     &                 rnonreg_targ_not_met_by_s333)
                      targ_not_met_s12abc=amax1((frac_targs12_21m+
     &                 frac_targs12_22m+frac_targs12_23m)*(rfcfs12(iseas)*
     &                 frf+rnonreg_targ_not_met_by_s333)-total_flow_s12,
     &                 0.0)
                      IF (trig_wca_stg(iup_wca).gt.regd) THEN
                        qs12t=rfcfs12(iseas)*frf+rfcfs12_reg(iseas)*(totf-
     &                   frf)+rmflws333_temp
                        IF (conv.eq.1.) THEN
                          s12_flow_targ=qs12t-rmflws333
                        END IF
                        s12qmax=total_frac*frf+total_frac_reg*(totf-frf)
                        qs12t=amin1(qs12t,s12qmax)
                        qs12=frac_targs12_24m*qs12t+rmflws12c
                        rnon_reg_comp=frac_targs12_24m*(rfcfs12(iseas)*
     &                   frf+add_nonreg_s12)+amin1(rmflws12c,
     &                   targ_not_met_s12abc)
                      ELSE
                        IF (frf.lt.250.0) THEN
                          ftargs12_low=frac_targs12_14m
                        ELSE
                          ftargs12_low=frac_targs12_24m
                        END IF
                        IF (nint(frf).gt.0) THEN
                          ratio=(avvolws(iwca3a,iconvey_c)+
     &                     sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                        ELSE
                          ratio=1.0
                        END IF
C
                        ratio=amin1(ratio,1.0)
C
                        IF (iprorte(iwca3a).eq.1) THEN
C
                          qs12t=rfcfs12(iseas)*frf*ratio+rmflws333_temp
                          rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*
     &                     ratio*frf+add_nonreg_s12)+amin1(rmflws12c,
     &                     targ_not_met_s12abc)
                        ELSE
                          qs12t=rfcfs12(iseas)*frf+rmflws333_temp
                          qs12t=amin1(qs12t,total_frac*frf,avvolws(iwca3a,
     &                     iconvey_c))
                          rnon_reg_comp=ftargs12_low*(rfcfs12(iseas)*frf+
     &                     add_nonreg_s12)+amin1(rmflws12c,
     &                     targ_not_met_s12abc)
                          rnon_reg_comp=amin1(rnon_reg_comp,
     &                     avvolws(iwca3a,iconvey_c))
                        END IF
                        qs12t=amin1(qs12t,total_frac*frf)
                        qs12=ftargs12_low*qs12t+rmflws12c
                        IF (conv.eq.1.) THEN
```

```
                    s12_flow_targ=rfcfs12(iseas)*frf
                  END IF
                END IF
                s12flow=amin1(qs12,s12cap)
                IF (iprorte(iwca3a).ne.1) THEN
                  s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                END IF
                s12_reg=amax1(s12flow-rnon_reg_comp,0.0)
                s12_env=amin1(s12flow,rnon_reg_comp)
              ELSE
                rnonreg_targ_not_met_by_s333=amax1(srsdmnd_e(jday)-
     &           kflo(ifls355)-kflo(ifls333),0.0)
                rnonreg_targ_not_met_by_s333=
     &           amin1(rnonreg_targ_not_met_by_s333,rmflws333_temp)
                IF (nint(totf_w).gt.0) THEN
                  ratio=(avvolws(iwca3a,iconvey_c)+
     &             sta34_out_to_wca3ane)/(rem_wsneed_s151+totf_w+
     &             totf_e)
                ELSE
                  ratio=1.0
                END IF
C
                ratio=amin1(ratio,1.0)
                IF (iprorte(iwca3a).eq.1) THEN
                  qs12t=totf_w*ratio+rmflws333_temp
                  rnon_reg_comp=frac_targs12_24m*(srsdmnd_w(jday)*
     &             ratio+rnonreg_targ_not_met_by_s333)
                ELSE
                  qs12t=totf_w+rmflws333_temp
                  qs12t=amin1(qs12t,avvolws(iwca3a,iconvey_c))
                  rnon_reg_comp=frac_targs12_24m*(srsdmnd_w(jday)+
     &             rnonreg_targ_not_met_by_s333)
                  rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iwca3a,
     &             iconvey_c))
                  ratio=1.0
                END IF
                qs12t=amin1(qs12t,total_frac*(totf_w+totf_e))
                qs12t=amax1(qs12t,totf_w*ratio)
                qs12=frac_targs12_24m*qs12t+rmflws12c
                s12flow=amin1(qs12,s12cap)
                IF (iprorte(iwca3a).ne.1) THEN
                  s12flow=amin1(s12flow,avvolws(iwca3a,iconvey_c))
                END IF
                targ_not_met_s12abc=(frac_targs12_21m+
     &           frac_targs12_22m+frac_targs12_23m)*(srsdmnd_w(jday)
     &           +rnonreg_targ_not_met_by_s333)-total_flow_s12
                targ_not_met_s12abc=amax1(targ_not_met_s12abc,0.0)
                rnon_reg_comp_1=rnon_reg_comp+amin1(rmflws12c,
     &           targ_not_met_s12abc)
                s12_reg=amax1(s12flow-rnon_reg_comp_1,0.0)
                s12_env=amin1(s12flow,rnon_reg_comp_1)
                IF (conv.eq.1.) THEN
                  s12_flow_targ=totf_w
                END IF
              END IF
              down_env_need=qs12
              IF (conv.eq.1.) THEN
```

```
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            ELSE
              qs12=s12cap
              IF (lwslo(iwenp)) THEN
                s12_env=qs12
                s12_reg=0.0
                down_env_need=qs12
              ELSE
                s12_env=0.0
                s12_reg=qs12
                down_env_need=0.0
              END IF
              IF (conv.eq.1.) THEN
                s12_reg_total=s12_reg+s12_reg_total
                s12_env_total=s12_env+s12_env_total
              END IF
            END IF
          END IF
          cout=amin1(qs12,s12cap)
        ELSE
          IF (stg3a28.gt.9.5) THEN
            qs12rg=2667.*(stg3a28-9.5)+1600.
          ELSE IF (stg3a28.gt.8.5) THEN
            qs12rg=1100.*(stg3a28-8.5)+500.
          ELSE IF (stg3a28.gt.7.5) THEN
            qs12rg=500.*(stg3a28-7.5)
          ELSE
            qs12rg=0.0
          END IF
          qs12rg_tot=qs12rg
          qs12rg=qs12rg*frac_targs12_24m
          IF (trig_wca_stg(iup_wca).gt.regstg_wca(iup_wca,1)) THEN
            qs12=s12cap
            cout=amin1(qs12,qs12rg)
            IF (qs12rg_tot.lt.qmdel) THEN
              qs12mindel=frac_targs12_24m*qmdel
              cout=amin1(qs12mindel,qs12)
            END IF
          ELSE
            qs12=frac_targs12_24m*(qmdel+rmflws333_temp)+rmflws12c
            cout=amin1(qs12,s12cap)
          END IF
          down_env_need=frac_targs12_24m*(qmdel+rmflws333_temp)+
&          rmflws12c
        END IF
        IF (trig_wca_stg(iup_wca).lt.regstg_wca(iup_wca,
&       itop_zone_indx(iup_wca+1)).or.wcaenv(idn_wca_indx(is8bsn-6)
&       ).eq.'TRUE ') THEN
          IF (frac_targs12_24m.lt.0.0001) THEN
            cout=0.0
          END IF
        END IF
        rmflws12d=qs12-cout
      ELSE
        cout=min(s12_capac,avvol_dn)
```

```
            IF (conv.eq.1.) THEN
              s12_flow_targ=cout+s12_flow_targ
            END IF
            rmflws12d=0.0
          END IF
C
          IF (.not.lflwtrg(ids_trg_idx(n,ind1))) THEN
            CALL match (flnm(iflo), 1, 6, struc_name_w_targ,
     &       no_of_struc_w_ds_targ_no_res, 6, istruc_targ_indx)
            cout=cout*frc_flow_targ(istruc_targ_indx)
          END IF
          total_flow_s12=total_flow_s12+cout
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            rlocal_flow=0.0
            avvol_up=-901.
C
            kflo(iflo)=cout
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s12cap,avvol_up,cout+rlocal_flow,down_env_need
            END IF
          END IF
C-------------------------------------------------------
C                S-355
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(50)) THEN
C
          iwca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s355cap=discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,ind1,1,1)
          ELSE
            s355cap=0.0
          END IF
          s355mx=amin1(s355cap,strdcap(n,ind1,1,1))
          stg_min_up=rmnstgs(n,ind1,iseas)
          avolsw=(chdep(n)-stg_min_up)*cl(n)*width(n)+cgseep(n,
     &     stg_min_up,0,0.)+tovflc(n,stg_min_up,0,0.,iter,1)+qu(n)
          avolsw=max(avolsw,0.0)/86400.
C
          iconvey_c=iconvey_c_s151
          floor=floor_elev(iwca3a,iconvey_c)
          cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &     cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
          aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &     tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,2)
          avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
```

```
     &     -outflow_from_3a_temp(iconvey_c)
          avvol_tot=max(avvol_tot,0.0)
          avvolws(iwca,iconvey_c)=avvol_tot
C
          IF (irceive(n,ind1).eq.1) THEN
            npts2=0
            cstor=0.0
            DO in=1,idncl-1
              npts2=npts2+nodcr(in)
            END DO
C
            rfin(idncl)=0.0
            etout(idncl)=0.0
            DO ic=1,nodcr(idncl)
              npts2=npts2+1
              nodec(ic)=nodecn(npts2)/10
              ilgth(ic)=mod(nodecn(npts2),10)
              etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/12.0
              rf=rain(nodec(ic))/(12.)
              rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*width(idncl)
              etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &         width(idncl)
            END DO
C
            IF (opt_mrp_to_enp) THEN
              IF (month.ge.1.and.month.le.6) THEN
                rmax_ds_cnl_stg=stg_min_fc(n,ind1,1)
              ELSE
                rmax_ds_cnl_stg=cnlstg(n,ind1,1)
              END IF
              swl_up_max=0.75*chdep(n)+0.25*tw
              swl=min(swl_up_max,rmax_ds_cnl_stg)
            ELSE
              swl=0.75*chdep(n)+0.25*tw
            END IF
C
          ELSE
            avvol_dn=0.0
            DO i=1,ncells_rec(n,ind1)
              gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
     &         s(nodrcv(n,ind1,i))
              IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
                tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
     &           ells(nodrcv(n,ind1,i))
              ELSE
                tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
     &           s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
              END IF
              swl_max=hw
              gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
              gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
              avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
     &         max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
     &         gw_dpth)*gdar
              avvol_dn=amax1(avvol,0.0)/86400.+avvol_dn
            END DO
          END IF
```

```
      IF (irceive(n,ind1).eq.1) THEN
        avail_cstor=(swl-chdep(idncl))*cl(idncl)*width(idncl)
        avail_cstor=avail_cstor-cgseep(idncl,swl,0,0.)-tovflc(idncl,
     &   swl,0,0.,iter,2)-rfin(idncl)+etout(idncl)
        avvol_dn=max(avail_cstor,0.0)/86400.
      END IF
      s355mx=min(s355mx,avvol_dn,avolsw)
      dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
      IF (pond(node_g3273).gt.dpthgw) THEN
        stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
      ELSE
        stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
     &   ells(node_g3273)
      END IF
C
      IF (opt_mrp_to_enp) THEN
        IF (irceive(n,ind1).eq.1) THEN
          IF (month.ge.1.and.month.le.6) THEN
            rmax_ds_cnl_stg=stg_min_fc(n,ind1,1)
          ELSE
            rmax_ds_cnl_stg=cnlstg(n,ind1,1)
          END IF
          IF (chdep(idncl).le.rmax_ds_cnl_stg-0.25) THEN
            frac_max_disc_c=1.0
          ELSE IF (chdep(idncl).lt.rmax_ds_cnl_stg) THEN
            frac_max_disc_c=(rmax_ds_cnl_stg-chdep(idncl))**0.50/
     &       (0.25**0.50)
          ELSE
            frac_max_disc_c=0.0
          END IF
        ELSE
          frac_max_disc_c=1.0
        END IF
        IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
          frac_max_disc_gc=0.0
        ELSE
          frac_max_disc_gc=1.0
        END IF
        s355mx=amin1(frac_max_disc_c,frac_max_disc_gc)*s355mx
      END IF
      IF (type_flow_across_ttrail.eq.'STAGETG') THEN
        depth_below_target=stage_import_wca(inesrs)-offset(inesrs,1)
     &   -avg_sim_wca_stg(inesrs)
        depth_below_target=max(depth_below_target,0.0)
        frac_cap=min(depth_below_target/0.1,1.0)
      ELSE
        frac_cap=1.0
      END IF
      s355mx=s355mx*frac_cap
      IF (wcaenv(iwca).eq.'FALSE') THEN
        IF (trig_wca_stg(iwca).gt.regstg_wca(iwca,1)) THEN
          q=s355mx
          IF (lwslo(inesrs)) THEN
            s355_env=q
            s355_reg=0.
            down_env_need=q
            s355_flow_targ=min(s355cap,avolsw,avvol_dn)
```

```
            ELSE
              IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                IF (iprorte(iwca3a).eq.1) THEN
                  IF (nint(frf).gt.0) THEN
                    ratio=(avvolws(iwca3a,iconvey_c)+
     &               sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                  ELSE
                    ratio=1.0
                  END IF
                ELSE
                  ratio=1.0
                END IF
                ratio=amin1(ratio,1.0)
                IF (type_target.eq.'TOTAL') THEN
                  down_env_need=rfcfs355(iseas)*frf*ratio-
     &             kflo(ifls333)
                ELSE
                  down_env_need=srsdmnd_e(jday)*ratio-kflo(ifls333)
                END IF
                down_env_need=amax1(down_env_need,0.0)
                s355_reg=amax1(q-down_env_need,0.0)
                s355_env=amin1(down_env_need,q)
                s355_flow_targ=min(s355cap,avolsw,avvol_dn)
                IF (trig_wca_stg(iwca).le.regstg_wca(iwca,
     &           itop_zone_indx(iwca+1))) THEN
                  flow_targ=rfcfs355(iseas)*totf-kflo(ifls333)
                  s355_flow_targ=amax1(s355_flow_targ,flow_targ)
                END IF
C
              ELSE
                s355_env=0.
                s355_reg=q
                down_env_need=0.0
                s355_flow_targ=0.0
              END IF
            END IF
          ELSE
            IF (type_flow_across_ttrail.ne.'STAGETG') THEN
              IF (iprorte(iwca3a).eq.1) THEN
                IF (nint(frf).gt.0) THEN
                  ratio=(avvolws(iwca3a,iconvey_c)+
     &             sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                ELSE
                  ratio=1.0
                END IF
              ELSE
                ratio=1.0
              END IF
              ratio=amin1(ratio,1.0)
              IF (type_target.eq.'TOTAL') THEN
                qtotf=rfcfs355(iseas)*frf*ratio+rfcfs355_reg(iseas)*
     &           (totf-frf)*ratio-kflo(ifls333)
                rnon_reg_comp=rfcfs355(iseas)*frf*ratio-kflo(ifls333)
                s355_flow_targ=rfcfs355(iseas)*frf+rfcfs355_reg(iseas)
     &           *(totf-frf)-kflo(ifls333)
              ELSE
                qtotf=totf_e*ratio-kflo(ifls333)
```

```
              rnon_reg_comp=amin1(srsdmnd_e(jday)*ratio,qtotf)-
&              kflo(ifls333)
              s355_flow_targ=totf_e-kflo(ifls333)
            END IF
            rnon_reg_comp=amax1(rnon_reg_comp,0.0)
            qtotf=amax1(qtotf,0.0)
            down_env_need=qtotf
            q=amin1(qtotf,s355mx)
            s355_reg=amax1(q-rnon_reg_comp,0.0)
            s355_env=amin1(q,rnon_reg_comp)
          ELSE
            IF (lwslo(inesrs)) THEN
              q=s355mx
            ELSE
              q=0.0
            END IF
            s355_reg=0.
            s355_env=q
            down_env_need=q
          END IF
        END IF
      ELSE
        IF (type_flow_across_ttrail.ne.'STAGETG') THEN
          IF (iprorte(iwca3a).eq.1) THEN
            IF (nint(frf).gt.0) THEN
              ratio=(avvolws(iwca3a,iconvey_c)+sta34_out_to_wca3ane)
&              /(rem_wsneed_s151+frf)
            ELSE
              ratio=1.0
            END IF
          ELSE
            ratio=1.0
          END IF
          ratio=amin1(ratio,1.0)
          IF (type_target.eq.'TOTAL') THEN
            qtotf=rfcfs355(iseas)*frf*ratio+rfcfs355_reg(iseas)*
&            (totf-frf)*ratio-kflo(ifls333)
            rnon_reg_comp=rfcfs355(iseas)*frf*ratio-kflo(ifls333)
            s355_flow_targ=rfcfs355(iseas)*frf+rfcfs355_reg(iseas)*
&            (totf-frf)-kflo(ifls333)
          ELSE
            qtotf=totf_e*ratio-kflo(ifls333)
            rnon_reg_comp=amin1(srsdmnd_e(jday)*ratio,qtotf)-
&            kflo(ifls333)
            s355_flow_targ=totf_e-kflo(ifls333)
          END IF
          rnon_reg_comp=amax1(rnon_reg_comp,0.0)
          qtotf=amax1(qtotf,0.0)
          down_env_need=qtotf
          q=amin1(qtotf,s355mx)
          s355_reg=amax1(q-rnon_reg_comp,0.0)
          s355_env=amin1(q,rnon_reg_comp)
        ELSE
          IF (lwslo(inesrs)) THEN
            q=s355mx
          ELSE
            q=0.0
```

```
              END IF
              s355_reg=0.
              s355_env=q
              down_env_need=q
            END IF
          END IF
          IF (type_flow_across_ttrail.ne.'STAGETG') THEN
            IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
              s355_avail_cap=min(s355cap,avolsw,avvol_dn)
              s355_avail_cap=min(s355_avail_cap,strdcap(n,ind1,1,1))-
     &         q
              s355_flw_targ=frac_reg_flow_nesrs_s355*(totf-frf)
              qadd=amin1(s355_avail_cap,s355_flw_targ)
              q=qadd+q
              s355_reg=qadd+s355_reg
            END IF
          END IF
          cout=q
          rmflws333_temp=rmflws333-amin1(cout,rmflws333)
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
            IF (nint(rmflws333).gt.0) rmflws333=rmflws333-amin1(cout,
     &       rmflws333)
            rlocal_flow=0.0
            avvol_up=-901.
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s355mx,avvol_up,kflo(iflo)+rlocal_flow,down_env_need
            END IF
          END IF

C-----------------------------------------------------------
C                S-337FC
C-----------------------------------------------------------
        ELSE IF (flnm(iflo).eq.'S337FC') THEN
          IF (nstorm_day(ndays_sim_sd).ne.1) THEN
C
C    compute regulatory flow from WCA-3A to be routed through S-337
C    to help relieve high water in WCA-3A and help the Sparrow
C    in western Shark River Slough
C
            tw_for_constrnt=0.0
            DO k=1,no_cells_trig_s12s343(2)
              tw_for_constrnt=h(icell_loc_trig_s12s343(2))+
     &         pond(icell_loc_trig_s12s343(2))+tw_for_constrnt
            END DO
            tw_for_constrnt_s12a=tw_for_constrnt/
```

```
     &       no_cells_trig_s12s343(2)
C
             tw_for_constrnt=0.0
             DO k=1,no_cells_trig_s12s343(3)
               tw_for_constrnt=h(icell_loc_trig_s12s343(3))+
     &          pond(icell_loc_trig_s12s343(3))+tw_for_constrnt
             END DO
             tw_for_constrnt_s12b=tw_for_constrnt/
     &       no_cells_trig_s12s343(3)
C
             tw_for_constrnt=0.0
             DO k=1,no_cells_trig_s12s343(4)
               tw_for_constrnt=h(icell_loc_trig_s12s343(4))+
     &          pond(icell_loc_trig_s12s343(4))+tw_for_constrnt
             END DO
             tw_for_constrnt_s12c=tw_for_constrnt/
     &       no_cells_trig_s12s343(4)
C
             tw_for_constrnt=0.0
             DO k=1,no_cells_trig_s12s343(5)
               tw_for_constrnt=h(icell_loc_trig_s12s343(5))+
     &          pond(icell_loc_trig_s12s343(5))+tw_for_constrnt
             END DO
             tw_for_constrnt_s12d=tw_for_constrnt/
     &       no_cells_trig_s12s343(5)
C
             IF (month.eq.7.and.iday.gt.15) THEN
               frac_targs12_21m=ftargs12(2,1,month+1)
               frac_targs12_22m=ftargs12(2,2,month+1)
               frac_targs12_23m=ftargs12(2,3,month+1)
               frac_targs12_24m=ftargs12(2,4,month+1)
               frac_targs12_31m=ftargs12(3,1,month+1)
               frac_targs12_32m=ftargs12(3,2,month+1)
               frac_targs12_33m=ftargs12(3,3,month+1)
               frac_targs12_34m=ftargs12(3,4,month+1)
             ELSE
               frac_targs12_21m=ftargs12(2,1,month)
               frac_targs12_22m=ftargs12(2,2,month)
               frac_targs12_23m=ftargs12(2,3,month)
               frac_targs12_24m=ftargs12(2,4,month)
               frac_targs12_31m=ftargs12(3,1,month)
               frac_targs12_32m=ftargs12(3,2,month)
               frac_targs12_33m=ftargs12(3,3,month)
               frac_targs12_34m=ftargs12(3,4,month)
             END IF
             lws12ac=tw_for_constrnt_s12a.gt.rmax_stg_s12s343(2,month)
     &       .or.frac_targs12_21m.lt.0.01.or.frac_targs12_31m.lt.0.01
             lws12bc=tw_for_constrnt_s12b.gt.rmax_stg_s12s343(3,month)
     &       .or.frac_targs12_22m.lt.0.01.or.frac_targs12_32m.lt.0.01
             lws12cc=tw_for_constrnt_s12c.gt.rmax_stg_s12s343(4,month)
     &       .or.frac_targs12_23m.lt.0.01.or.frac_targs12_33m.lt.0.01
             lws12dc=tw_for_constrnt_s12d.gt.rmax_stg_s12s343(5,month)
     &       .or.frac_targs12_24m.lt.0.01.or.frac_targs12_34m.lt.0.01
             IF (iopt_for_reg_releases_s334.eq.5.or.(iopt_for_reg_release
     &       s_s334.eq.1.and.lws12ac)
     &       .or.(iopt_for_reg_releases_s334.eq.2.and.lws12ac.and.lws12b
     &       c)
```

```
     &       .or.(iopt_for_reg_releases_s334.eq.3.and.lws12ac.and.lws12b
     &       c.and.lws12cc)
     &       .or.(iopt_for_reg_releases_s334.eq.4.and.lws12ac.and.lws12b
     &       c.and.lws12cc.and.lws12dc)) THEN
              is335io = 0
              DO jj=1,nstrctr(l30)
                 IF (flnm(istr(l30,jj)).eq.'S335IO') is335io = 1
              ENDDO
              DO jj=1,nstrctr(l30)
                IF (flnm(istr(l30,jj)).eq.'S335' .and.
     &           is335io .eq. 0) kstr_no=jj
                IF (is335io .eq. 1 .and.  flnm(istr(l30,jj))
     &          .eq.'S335IO') kstr_no=jj
              END DO
              DO nn=1,nch
                DO kk=1,nstrctr(nn)
                  IF (flnm(istr(nn,kk)).eq.'COMBQ') THEN
                    istr_s331_no=kk
                    icanal_no_s331=nn
                    GO TO 30
                  END IF
                END DO
              END DO
   30         tot_pump=0.0
Ccbwnewline1-24-01
              dischg_from_l30=strout(chdep(l30),l30,l30,kstr_no,0,crmin,
     &         volwtr,prevqu,rain,iter,tot_pump,jday,tempf_volwtr,
     &         tempf_weircff,0) - s337fc
              dischg_from_l30 = max(dischg_from_l30,0.0)
c
              if (l29 .gt. l30) then
                 vol_dn_s334 = prev_voldn_s334
              else
                 vol_dn_s334 = avvol_dns334
              endif
              avvol_dns = amax1(vol_dn_s334 - s334_fc,0.0)
              rem_capac_s331=discoef(icanal_no_s331,istr_s331_no,2,imgt)
     &         *frac_struc_cap(icanal_no_s331,istr_s331_no,month)-
     &         dischg_from_l30-s334_fc
              rem_capac_s331=amax1(rem_capac_s331,0.0)
              avail_cap_s337_fc=capacfl(imgt,1)
              cout=amin1(avail_cap_s337_fc,rem_capac_s331,avvol_dns,
     &         amax1(regs151-regs31,0.0))
              IF (conv.eq.1) THEN
                IF (irceive(n,ind1).eq.1) THEN
                  qu(idncl)=qu(idncl)+cout*86400.
                ELSE
                  DO i=1,ncells_rec(n,ind1)
                    pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &               86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &               (nodrcv(n,ind1,i))-total_small_afact
     &               (nodrcv(n,ind1,i))))
                  END DO
                END IF
                kflo(iflo)=cout
                s337fc=cout
              END IF
```

```
            ELSE
              cout=0.0
              kflo(iflo)=0.0
              s337fc=0.0
            END IF
          ELSE
            cout=0.0
            kflo(iflo)=0.0
            s337fc=0.0
          END IF
c           if (conv.eq.1.) print *,capacfl(imgt,1)
c    $              ,dischg_from_l30,s333_reg_lec
c    $              ,cout,regs151,regs31,'  S337FC  '
c    $              ,rem_capac_s331,avvol_dns,s334_fc
c    $              ,chdep(l30),chdep(l31nc)
C-----------------------------------------------------------
C                S349A
C-----------------------------------------------------------
        ELSE IF (flnm(iflo).eq. 'S349A ') THEN
c
C
          IF (chdep(n).gt.tw) THEN
            s349amx=discoef(n,ind1,1,imgt)*(chdep(n)-tw)**pwr(n,ind1,1,
     &       imgt)
          ELSE
            s349amx=0.0
          END IF
          CALL match (cnm(n), 1, 5, convey_canal_names, nconvey_total,
     &     5, imatch)
          DO isa=1,nwcas
            IF (imatch.le.n_conv_tot(isa)) THEN
              IF (isa.gt.1) THEN
                iconvey_c=imatch-n_conv_tot(isa-1)
              ELSE
                iconvey_c=imatch
              END IF
              GO TO 71
            END IF
          END DO
 71      cout = amin1(s333_ws,s349amx)
c        if (conv.eq.1.) print *,wsdel,s349amx
c    &    ,avvolws(iup_wca,iconvey_c),s333_ws,'  CVCVCVCV  '
c    &    ,chdep(n),tw
         IF (conv.eq.1.) THEN
           avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,iconvey_c)-
     &      cout
           outflow_from_3a(iconvey_c)=cout  +
     &      outflow_from_3a(iconvey_c)
           IF (irceive(n,ind1).eq.1) THEN
             qu(idncl)=qu(idncl)+cout*86400.
           ELSE
             DO i=1,ncells_rec(n,ind1)
               pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &          86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &          (nodrcv(n,ind1,i))-total_small_afact
     &          (nodrcv(n,ind1,i))))
             END DO
```

```
           END IF
           kflo(iflo)=cout
         END IF
C-------------------------------------------------------
C                S349B
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq. 'S349B ') THEN
           cout  = min(capacws(imgt),s333_ws)
           IF (conv.eq.1.) THEN
             IF (irceive(n,ind1).eq.1) THEN
               qu(idncl)=qu(idncl)+cout*86400.
             ELSE
               DO i=1,ncells_rec(n,ind1)
                 pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &            86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &            (nodrcv(n,ind1,i))-total_small_afact
     &            (nodrcv(n,ind1,i))))
               END DO
             END IF
             kflo(iflo)=cout
         END IF
C-------------------------------------------------------
C                S349C
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq. 'S349C ') THEN
           cout = min(capacws(imgt),s333_ws)
           IF (conv.eq.1.) THEN
             IF (irceive(n,ind1).eq.1) THEN
               qu(idncl)=qu(idncl)+cout*86400.
             ELSE
               DO i=1,ncells_rec(n,ind1)
                 pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &            86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &            (nodrcv(n,ind1,i))-total_small_afact
     &            (nodrcv(n,ind1,i))))
               END DO
             END IF
             kflo(iflo)=cout
         END IF
c
C
C-------------------------------------------------------
C                S-333
C-------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(51)) THEN
C
          iup_wca=idn_wca_indx(is8bsn-6)
C
          IF (chdep(n).gt.tw) THEN
            s333mx=discoef(n,ind1,1,imgt)*(chdep(n)-tw)**pwr(n,ind1,1,
     &       imgt)
          ELSE
            s333mx=0.0
          END IF
          s333_max_cap=s333mx
          total_des_cap=strdcap(n,ind1,1,1)
```

```
          IF (nsubstr(n,ind1).gt.1) THEN
            DO kk=2,nsubstr(n,ind1)
              s333mx=s333mx+discoef(n,ind1,kk,imgt)
              s333_max_cap=s333_max_cap+discoef(n,ind1,kk,imgt)
              total_des_cap=total_des_cap+strdcap(n,ind1,kk,1)
            END DO
          END IF
          s333mx=amin1(s333mx,total_des_cap)
          ws_need_l31nc_orig=dqu(l31nc)
          is349a=0
          do ii = 1,nstrctr(ica3)
            IF (tmpname(ica3,ii).eq.'S349A ') is349a=1
          enddo
          IF (is349a.eq.0) THEN
            CALL match (cnm(n), 1, 5, convey_canal_names, nconvey_total,
     &       5, imatch)
            DO isa=1,nwcas
              IF (imatch.le.n_conv_tot(isa)) THEN
                IF (isa.gt.1) THEN
                  iconvey_c=imatch-n_conv_tot(isa-1)
                ELSE
                  iconvey_c=imatch
                END IF
                GO TO 40
              END IF
            END DO
          ELSE
            iconvey_c=iconvey_c_s151
          ENDIF
C
C  40     if (is349a.eq.0) THEN
   40      floor=floor_elev(iwca3a,iconvey_c)
           cnlstor=(chdep(intcnl(iwca3a,iconvey_c))-floor)*
     &     cl(intcnl(iwca3a,iconvey_c))*width(intcnl(iwca3a,iconvey_c))
           aval_vol=cnlstor+cgseep(intcnl(iwca3a,iconvey_c),floor,0,0.)+
     &     tovflc(intcnl(iwca3a,iconvey_c),floor,0,0.,iter,1)
           avvol_tot=max(aval_vol,0.0)/86400.+tinflow(iwca3a,iconvey_c)
     &     -outflow_from_3a_temp(iconvey_c)
           avvol_tot=max(avvol_tot,0.0)
           avvolws(iup_wca,iconvey_c)=avvol_tot
           IF (dqu(l31nc).gt.0.0.and.use_enp_ws_to_lec.eq.'TRUE '
     &     .and.irceive(n,ind1).eq.1) THEN
C
            DO nn=1,nch
              DO ii=1,nstrctr(nn)
                IF (flnm(istr(nn,ii)).eq.strname_spec(40)) THEN
                  IF (chdep(nn)+0.3*hdc(n,iseas).gt.chdep(idncnl(nn,ii)
     &              )) THEN
                    add_capac_ws=discoef(nn,ii,1,1)*(chdep(nn)+0.3*hdc(
     &               nn,iseas)-chdep(idncnl(nn,ii)))**pwr(nn,ii,1,1)
                  ELSE
                    add_capac_ws=0.0
                  END IF
                  is151_sim_ws=1
                  GO TO 50
                ELSE
                  add_capac_ws=0.0
```

```
                is151_sim_ws=0
              END IF
            END DO
          END DO
   50     IF (is151_sim_ws.eq.0) THEN
            DO nn=1,nch
              DO ii=1,nstrctr(nn)
                IF (flnm(istr(nn,ii)).eq.strname_spec(7)) THEN
                  IF (chdep(nn).gt.chdep(idncnl(nn,ii))) THEN
                    add_capac_ws=discoef(nn,ii,1,1)*(chdep(nn)-
     &               chdep(idncnl(nn,ii)))**pwr(nn,ii,1,1)
                  ELSE
                    add_capac_ws=0.0
                  END IF
                  is151_sim_ws=1
                  GO TO 60
                ELSE
                  add_capac_ws=0.0
                  is151_sim_ws=0
                END IF
              END DO
            END DO
          END IF
   60     tot_capac=s333mx+add_capac_ws
          IF (tot_capac.gt.0.0) THEN
            ratio_cap=s333mx/tot_capac
          ELSE
            ratio_cap=0.0
          END IF
C
          IF (iprorte(iwca3a).eq.1) THEN
            tneed_s333=dqu(l30)+dqu(icnl_dn_s31)
            ratio=avvolws(iup_wca,iconvey_c)/tneed_s333
            ratio=amin1(ratio,1.0)
            wsdel=dqu(l31nc)*ratio*ratio_cap*frac_del_frm_wca(iwca3a)
          ELSE
            wsdel=amin1(dqu(l31nc),avvolws(iup_wca,iconvey_c))*
     &       ratio_cap*frac_del_frm_wca(iwca3a)
          END IF
          IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
            rega=regstg_wca(iwca3a,itop_zone_indx(iwca3a+1))
            IF (trig_wca_stg(iwca3a).le.rega) THEN
              qs333env=rfcfs355(iseas)*frf+rfcfs355_reg(iseas)*(totf-
     &         frf)
              qs333ws=s333mx-(qs333env-kflo(ifls355))
              qs333ws=amax1(qs333ws,0.0)
C
              qs333ws=amin1(qs333ws,wsdel)*ratflw

              kflo(iflo)=amin1(qs333ws,s333mx)
            ELSE
              q333ws=amin1(s333mx,wsdel)
              kflo(iflo)=amax1(q333ws,0.)
            END IF
            IF (conv.eq.1. .and. is349a .eq.0) THEN
              avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,iconvey_c)-
     &         kflo(iflo)
```

```
                outflow_from_3a(iconvey_c)=kflo(iflo)+
     &           outflow_from_3a(iconvey_c)
              END IF
            ELSE
              q=amin1(s333mx,wsdel)
              kflo(iflo)=amax1(q,0.)
              IF (conv.eq.1. .and. is349a .eq.0) THEN
                avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,iconvey_c)-
     &           kflo(iflo)
                outflow_from_3a(iconvey_c)=kflo(iflo)+
     &           outflow_from_3a(iconvey_c)
              END IF
              down_env_need=dqu(l31nc)
C
            END IF
          ELSE
            kflo(iflo)=0.
            down_env_need=0.0
          END IF
          cout=kflo(iflo)
          s333_ws=cout
C
          tdqu_dn(l31nc)=dqu(l31nc)-kflo(iflo)
          IF (conv.ne.1. .and. is349a .eq. 0) THEN
            outflow_from_3a_temp(iconvey_c)=kflo(iflo)+
     &       outflow_from_3a(iconvey_c)
            upstream_inflow(l31nc)=upstream_inflow(l31nc)+kflo(iflo)*
     &       86400.
          ELSE
            outflow_from_3a_temp(iconvey_c)=outflow_from_3a(iconvey_c)
          END IF
          icnl_dns=l31nc
   70     icnl_dns=iup_canal_ws(icnl_dns)
          IF (icnl_dns.le.nch) THEN
            tdqu_dn(icnl_dns)=dqu(icnl_dns)-kflo(iflo)
            tdqu_dn(icnl_dns)=amax1(tdqu_dn(icnl_dns),0.0)
            GO TO 70
          END IF
C
          s333_max_up=s333mx-kflo(iflo)
          s333_max_cap_up=s333_max_cap-kflo(iflo)
          IF (ipurpse(n,ind1,1).ne.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              npts2=0
              cstor=0.0
              DO in=1,idncl-1
                npts2=npts2+nodcr(in)
              END DO
C
              rfin(idncl)=0.0
              etout(idncl)=0.0
              DO ic=1,nodcr(idncl)
                npts2=npts2+1
                nodec(ic)=nodecn(npts2)/10
                ilgth(ic)=mod(nodecn(npts2),10)
                etcn=kmax(lutyp(nodec(ic)))*potet(ietzon(nodec(ic)))/
     &           12.0
```

```
                rf=rain(nodec(ic))/(12.)
                rfin(idncl)=rfin(idncl)+rf*distnc(ilgth(ic))*
     &           width(idncl)
                etout(idncl)=etout(idncl)+etcn*distnc(ilgth(ic))*
     &           width(idncl)
              END DO
C
              IF (opt_mrp_to_enp) THEN
                IF (month.ge.1.and.month.le.6) THEN
                  rmax_ds_cnl_stg=stg_min_fc(n,ind1,1)
                ELSE
                  rmax_ds_cnl_stg=cnlstg(n,ind1,1)
                END IF
                swl=min(chdep(n),rmax_ds_cnl_stg)
              ELSE
                swl=chdep(n)
              END IF
C
            ELSE
              avvol_dn=0.0
              avvol_dn_max=0.0
              DO i=1,ncells_rec(n,ind1)
                gw_depth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
     &           s(nodrcv(n,ind1,i))
                IF (pond(nodrcv(n,ind1,i)).gt.gw_depth) THEN
                  tw_node_stg=pond(nodrcv(n,ind1,i))-gw_depth+
     &             ells(nodrcv(n,ind1,i))
                ELSE
                  tw_node_stg=(pond(nodrcv(n,ind1,i))-gw_depth)/
     &             s(nodrcv(n,ind1,i))+ells(nodrcv(n,ind1,i))
                END IF
                swl_max=hw
                gw_dpth=ells(nodrcv(n,ind1,i))-swl_max
                gw_dpth=amax1(gw_dpth,0.0)*s(nodrcv(n,ind1,i))
                avvol=(max(swl_max,ells(nodrcv(n,ind1,i)))-
     &           max(tw_node_stg,ells(nodrcv(n,ind1,i)))+gw_depth-
     &           gw_dpth)*gdar
                avvol_dn=amax1(avvol,0.0)/86400.+avvol_dn
                avvol_dn_max=avvol_dn
              END DO
            END IF
            IF (irceive(n,ind1).eq.1) THEN
              avail_cstor=(swl-chdep(idncl))*cl(idncl)*width(idncl)
              avail_cstor_max=(chdep(n)-chdep(idncl))*cl(idncl)*
     &          width(idncl)
              dischg=0.0
              DO is=1,nstrctr(idncl)
                dischg=strout(swl,idncl,idncl,is,0,crmin,volwtr,prevqu,
     &           rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)+
     &           dischg
              END DO
C
              avail_cstor=avail_cstor-cgseep(idncl,swl,0,0.)-
     &          tovflc(idncl,swl,0,0.,iter,2)-rfin(idncl)+etout(idncl)+
     &          dischg*86400.
              avail_cstor_max=avail_cstor_max-cgseep(idncl,chdep(n),0,
     &          0.)-tovflc(idncl,chdep(n),0,0.,iter,2)-rfin(idncl)+
```

```
     &         etout(idncl)+dischg*86400.
               avail_cstor=avail_cstor-(kflo(ifls356)+kflo(ifls355))*
     &         86400.
               avail_cstor_max=avail_cstor_max-(kflo(ifls356)+
     &         kflo(ifls355))*86400.
               avvol_dn=max(avail_cstor,0.0)/86400.
               avvol_dn_max=max(avail_cstor_max,0.0)/86400.
             END IF
             IF (type_flow_across_ttrail.eq.'STAGETG') THEN
               depth_below_target=stage_import_wca(inesrs)-offset(inesrs,
     &         1)-avg_sim_wca_stg(inesrs)
               depth_below_target=max(depth_below_target,0.0)
               frac_cap=min(depth_below_target/0.1,1.0)
               IF (avg_sim_wca_stg_out(igs333_enp)
     &         .gt.stage_outflw_wca(igs333_enp)+offset_out(igs333_enp,2)
     &         ) THEN
                 depth_above_sec_offset=avg_sim_wca_stg_out(igs333_enp)-
     &           stage_outflw_wca(igs333_enp)-offset_out(igs333_enp,2)
                 depth_above_sec_offset=amax1(depth_above_sec_offset,0.0)
                 frac_cap_up=amin1(depth_above_sec_offset/0.1,1.0)
                 IF (.not.lwslo(inesrs)) THEN
                   frac_cap_outflow=frac_cap_up
                 ELSE
                   frac_cap_outflow=amax1(frac_cap,frac_cap_up)
                 END IF
               ELSE IF (avg_sim_wca_stg_out(igs333_enp)
     &         .gt.stage_outflw_wca(igs333_enp)+offset_out(igs333_enp,1)
     &         ) THEN
                 depth_above_noact_zone=avg_sim_wca_stg_out(igs333_enp)-
     &           stage_outflw_wca(igs333_enp)-offset_out(igs333_enp,1)
                 depth_above_noact_zone=amax1(depth_above_noact_zone,0.0)
                 frac_cap_up=depth_above_noact_zone/(0.20*
     &           (offset_out(igs333_enp,2)-offset_out(igs333_enp,1)))
                 frac_cap_up=amin1(frac_cap_up,1.0)
                 frac_cap_outflow=amin1(frac_cap_up,frac_cap)
               ELSE
                 frac_cap_up=0.0
                 frac_cap_outflow=0.0
               END IF
             ELSE
               frac_cap_outflow=1.0
             END IF
             s333mx=min(s333_max_up,avvol_dn)*frac_cap_outflow
C            if (conv.eq.1.)
C    $       print *,avg_sim_wca_stg_out(igs333_enp),igs333_enp
C    $         ,frac_cap_up,frac_cap,offset_out(igs333_enp,1),'  S333OUT'
C    $         ,stage_outflw_wca(igs333_enp),offset_out(igs333_enp,2)
C    $         ,depth_below_target,LWSLO(inesrs)
C    $         ,stage_import_wca(inesrs),inesrs
C    $         ,avg_sim_wca_stg(inesrs),frac_cap_outflow,avvol_dn

             dpthgw=(ells(node_g3273)-h(node_g3273))*s(node_g3273)
             IF (pond(node_g3273).gt.dpthgw) THEN
               stage_g3273=pond(node_g3273)-dpthgw+ells(node_g3273)
             ELSE
               stage_g3273=(pond(node_g3273)-dpthgw)/s(node_g3273)+
     &         ells(node_g3273)
```

```
            END IF
C
            IF (opt_mrp_to_enp) THEN
              IF (irceive(n,ind1).eq.1) THEN
                IF (month.ge.1.and.month.le.6) THEN
                  rmax_ds_cnl_stg=stg_min_fc(n,ind1,1)
                ELSE
                  rmax_ds_cnl_stg=cnlstg(n,ind1,1)
                END IF
                IF (chdep(idncl).le.rmax_ds_cnl_stg-0.3) THEN
                  frac_max_disc_c=1.0
                ELSE IF (chdep(idncl).lt.rmax_ds_cnl_stg) THEN
                  frac_max_disc_c=(rmax_ds_cnl_stg-chdep(idncl))**0.33/
     &             (0.30**0.333)
                ELSE
                  frac_max_disc_c=0.0
                END IF
              ELSE
                frac_max_disc_c=1.0
              END IF
              IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
                frac_max_disc_gc=0.0
              ELSE
                frac_max_disc_gc=1.0
              END IF
              s333mx=amin1(frac_max_disc_c,frac_max_disc_gc)*s333mx
            END IF
C
            IF (type_flow_across_ttrail.ne.'MINDEL ') THEN
              IF (type_flow_across_ttrail.ne.'STAGETG') THEN
                IF (iprorte(iwca3a).eq.1) THEN
                  IF (nint(frf).gt.0) THEN
                    ratio=(avvolws(iup_wca,iconvey_c)+
     &               sta34_out_to_wca3ane)/(rem_wsneed_s151+frf)
                  ELSE
                    ratio=1.0
                  END IF
                ELSE
                  ratio=1.0
                END IF
                ratio=amin1(ratio,1.0)
                IF (trig_wca_stg(iup_wca).le.regstg_wca(iup_wca,
     &           itop_zone_indx(iup_wca+1))) THEN
                  IF (type_target.eq.'TOTAL') THEN
                    qs333=rfcfs355(iseas)*frf*ratio+rfcfs355_reg(iseas)*
     &               (totf-frf)*ratio-kflo(ifls355)
                    rnon_reg_comp=rfcfs355(iseas)*frf*ratio-
     &               kflo(ifls355)
                    IF (iprorte(iwca3a).ne.1) THEN
                      qs333=amin1(qs333,avvolws(iup_wca,iconvey_c))
                      rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iup_wca,
     &                 iconvey_c))
                      IF (conv.eq.1.) THEN
                        avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,
     &                   iconvey_c)-qs333
                      END IF
                    END IF
```

```
                  rnon_reg_comp=amax1(rnon_reg_comp,0.0)
                  s333_flow_targ=rfcfs355(iseas)*frf+
     &             rfcfs355_reg(iseas)+rfcfs355_reg(iseas)*(totf-frf)-
     &             kflo(ifls355)
                ELSE
                  qs333=totf_e*ratio-kflo(ifls355)
                  rnon_reg_comp=amax1(srsdmnd_e(jday)*ratio-
     &             kflo(ifls355),0.0)
                  IF (iprorte(iwca3a).ne.1) THEN
                    qs333=amin1(qs333,avvolws(iup_wca,iconvey_c))
                    rnon_reg_comp=amin1(rnon_reg_comp,avvolws(iup_wca,
     &               iconvey_c))
                    IF (conv.eq.1.) THEN
                      avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,
     &                 iconvey_c)-qs333
                    END IF
                  END IF
                  s333_flow_targ=totf_e-kflo(ifls355)
                END IF
C
                IF (qs333.lt.0.0) qs333=0.0
                down_env_need=qs333
                cout=amin1(s333mx,qs333)
                s333_reg=amax1(cout-rnon_reg_comp,0.0)
                s333_env=amin1(cout,rnon_reg_comp)
                rmflws333=amax1(qs333-cout,0.0)
              ELSE
                cout=s333mx
                rmflws333=0.0
                IF (type_target.eq.'TOTAL') THEN
                  down_env_need=amax1(rfcfs355(iseas)*frf*ratio-
     &             kflo(ifls355),0.0)
                ELSE
                  down_env_need=amax1(srsdmnd_e(jday)*ratio-
     &             kflo(ifls355),0.0)
                END IF
                IF (conv.eq.1.) THEN
                  avvolws(iup_wca,iconvey_c)=avvolws(iup_wca,
     &             iconvey_c)-cout
                END IF
                s333_reg=amax1(cout-down_env_need,0.0)
                s333_env=amin1(down_env_need,cout)
                s333_flow_targ=amin1(s333_max_cap_up,avvol_dn_max,
     &           total_des_cap)
              END IF
            ELSE
              cout=s333mx
              IF (lwslo(inesrs)) THEN
                s333_env=cout
                s333_reg=0.0
                down_env_need=s333_env
              ELSE
                s333_env=0.0
                s333_reg=cout
                down_env_need=0.0
              END IF
            END IF
```

```
              cout_total=cout+s333_ws
              cout=cout_total
            ELSE
              rmflws333=0.0
            END IF
          ELSE
            cout=s333_ws
          END IF
C
C   when stage at G3273 is above maximum allowable stage for S-333
C    flow into NESRS for normal operations,allow a fraction of the
C    regulatory flow (input by the user) to be routed to NESRS
C
C
          IF (type_flow_across_ttrail.ne.'STAGETG') THEN
            IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
              IF (trig_wca_stg(iwca3a).le.regstg_wca(iwca3a,
     &         itop_zone_indx(iwca3a+1))) THEN
               s333_flw_targ=frac_reg_flow_nesrs_s333*(totf-frf)-
     &          kflo(ifls355)
              ELSE
               IF (frac_reg_flow_nesrs_s333.gt.0.01) THEN
                 s333_flw_targ=999999.
               ELSE
                 s333_flw_targ=0.00
               END IF
              END IF
              avail_cap_s333=amin1(s333_max_cap,total_des_cap)*
     &         frac_max_disc_c-cout
              avail_cap_s333=amax1(avail_cap_s333,0.0)
              s333flw=amin1(s333_flw_targ,avail_cap_s333)
              IF (trig_wca_stg(iwca3a).le.regstg_wca(iwca3a,
     &         itop_zone_indx(iwca3a+1))) THEN
               rmflws333=qs333-s333flw
              END IF
              cout=s333flw+cout
              IF (conv.eq.1.) s333_reg=s333flw+s333_reg
              add_s333_above_const=s333flw
              ig3273_const_indx=1
            ELSE
              add_s333_above_const=0.0
              ig3273_const_indx=0
            END IF
          END IF
C
C   compute regulatory flow from WCA-3A to be routed through S-333
C    and S-334 to help relieve high water in WCA-3A and help the Sparrow
C    in western Shark River Slough
C
          tw_for_constrnt=0.0
          DO k=1,no_cells_trig_s12s343(2)
            tw_for_constrnt=h(icell_loc_trig_s12s343(2))+
     &       pond(icell_loc_trig_s12s343(2))+tw_for_constrnt
          END DO
          tw_for_constrnt_s12a=tw_for_constrnt/no_cells_trig_s12s343(2)
C
          tw_for_constrnt=0.0
```

```
      DO k=1,no_cells_trig_s12s343(3)
        tw_for_constrnt=h(icell_loc_trig_s12s343(3))+
     &   pond(icell_loc_trig_s12s343(3))+tw_for_constrnt
      END DO
      tw_for_constrnt_s12b=tw_for_constrnt/no_cells_trig_s12s343(3)
C
      tw_for_constrnt=0.0
      DO k=1,no_cells_trig_s12s343(4)
        tw_for_constrnt=h(icell_loc_trig_s12s343(4))+
     &   pond(icell_loc_trig_s12s343(4))+tw_for_constrnt
      END DO
      tw_for_constrnt_s12c=tw_for_constrnt/no_cells_trig_s12s343(4)
C
      tw_for_constrnt=0.0
      DO k=1,no_cells_trig_s12s343(5)
        tw_for_constrnt=h(icell_loc_trig_s12s343(5))+
     &   pond(icell_loc_trig_s12s343(5))+tw_for_constrnt
      END DO
      tw_for_constrnt_s12d=tw_for_constrnt/no_cells_trig_s12s343(5)
C
      IF (month.eq.7.and.iday.gt.15) THEN
        frac_targs12_21m=ftargs12(2,1,month+1)
        frac_targs12_22m=ftargs12(2,2,month+1)
        frac_targs12_23m=ftargs12(2,3,month+1)
        frac_targs12_24m=ftargs12(2,4,month+1)
        frac_targs12_31m=ftargs12(3,1,month+1)
        frac_targs12_32m=ftargs12(3,2,month+1)
        frac_targs12_33m=ftargs12(3,3,month+1)
        frac_targs12_34m=ftargs12(3,4,month+1)
      ELSE
        frac_targs12_21m=ftargs12(2,1,month)
        frac_targs12_22m=ftargs12(2,2,month)
        frac_targs12_23m=ftargs12(2,3,month)
        frac_targs12_24m=ftargs12(2,4,month)
        frac_targs12_31m=ftargs12(3,1,month)
        frac_targs12_32m=ftargs12(3,2,month)
        frac_targs12_33m=ftargs12(3,3,month)
        frac_targs12_34m=ftargs12(3,4,month)
      END IF
      lws12ac=tw_for_constrnt_s12a.gt.rmax_stg_s12s343(2,month)
     &  .or.frac_targs12_21m.lt.0.01.or.frac_targs12_31m.lt.0.01
      lws12bc=tw_for_constrnt_s12b.gt.rmax_stg_s12s343(3,month)
     &  .or.frac_targs12_22m.lt.0.01.or.frac_targs12_32m.lt.0.01
      lws12cc=tw_for_constrnt_s12c.gt.rmax_stg_s12s343(4,month)
     &  .or.frac_targs12_23m.lt.0.01.or.frac_targs12_33m.lt.0.01
      lws12dc=tw_for_constrnt_s12d.gt.rmax_stg_s12s343(5,month)
     &  .or.frac_targs12_24m.lt.0.01.or.frac_targs12_34m.lt.0.01
C
      IF (trig_wca_stg(iwca3a).ge.regstg_wca(iwca3a,
     &  itop_zone_indx(iwca3a+1))+s333s334rg_offset.or.l334rg) THEN
        l334rg=.true.
        l334rg=.not.(trig_wca_stg(iwca3a).lt.regstg_wca(iwca3a,
     &   itop_zone_indx(iwca3a+1)))
      END IF
      add_s333_fc_s334=0.0
      IF ((iopt_for_reg_releases_s334.eq.5.or.(iopt_for_reg_releases
     &  _s334.eq.1.and.lws12ac)
```

```
&     .or.(iopt_for_reg_releases_s334.eq.2.and.lws12ac.and.lws12bc)
&     .or.(iopt_for_reg_releases_s334.eq.3.and.lws12ac.and.lws12bc.
&     and.lws12cc)
&     .or.(iopt_for_reg_releases_s334.eq.4.and.lws12ac.and.lws12bc.
&     and.lws12cc.and.lws12dc)
&     .or.(iopt_for_reg_releases_s334.ne.0.and.l334rg))
&     .and.nstorm_day(ndays_sim_sd).ne.1) THEN
       IF (irceive(n,ind1).eq.1) THEN
         CALL match (canl_names_spec(47), 1, 5, cnm, nch+1, 5,
&         l29c)
         DO ii=1,nstrctr(l29c)
           IF (flnm(istr(l29c,ii)).eq.'S334FC') THEN
             IF (chdep(l29c).gt.chdep(idncnl(l29c,ii))+hdc(l29c,
&             iseas)) THEN
               capac_s334=discoef(l29c,ii,1,1)*(chdep(l29c)-
&               chdep(idncnl(l29c,ii))-hdc(l29c,iseas))**pwr(l29c,
&               ii,1,1)
               IF (ides_cap_limit(l29c,ii)
&               .eq.1.and.(stage_g3273.le.stg_trigger_fl(n,ind1,1)
&               .or.is334_open_flg.ne.0.or.frac_struc_cap(n,ind1,
&               month).eq.1.0)) THEN
                capac_s334=amin1(capac_s334,strdcap(l29c,ii,1,
&                imgt))
               END IF
               capac_s334=capac_s334*frac_struc_cap(l29c,ii,month)
             ELSE
               capac_s334=0.0
             END IF
             IF (ndsclim(l29c,ii).gt.0) THEN
               if (l29 .lt. l30) then
                 is335io = 0
                 DO jj=1,nstrctr(l30)
                    IF (flnm(istr(l30,jj)).eq.'S335IO') is335io = 1
                 ENDDO
                 DO jj=1,nstrctr(l30)
                   IF (flnm(istr(l30,jj)).eq.'S335' .and.
&                   is335io .eq. 0) kstr_no=jj
                   IF (is335io .eq. 1 .and.  flnm(istr(l30,jj))
&                   .eq.'S335IO') kstr_no=jj
                 END DO
                 tot_pump=0.0
                 dischg_from_l30=strout(chdep(l30),l30,l30,kstr_no
&                 ,0,crmin,volwtr,prevqu,rain,iter,tot_pump,jday
&                 ,tempf_volwtr,tempf_weircff,0) - s337fc
                 dischg_from_l30=max(dischg_from_l30,0.0)
               else
                 dischg_from_l30 = 0.0
               endif
               dischgt=0.0
               if (nodcr(idncnl(l29c,ii)) .gt. 1) then
                 stage_diff=float(nodcr(idncnl(l29c,ii))-
&                 nodepos_sd(l29c,ii))/float(nodcr(idncnl(l29c,ii))
&                 -1)*hdc(idncnl(l29c,ii),iseas)
               else
                 stage_diff=0.0
               endif
               DO kk=1,nstrctr(idncnl(l29c,ii))
```

```
                    if (nodcr(idncnl(l29c,ii)) .gt. 1) then
                      stagedff=float(nodcr(idncnl(l29c,ii))-
     &                nodepos_su(idncnl(l29c,ii),kk))/
     &                float(nodcr(idncnl(l29c,ii))-1)*hdc(idncnl(l29c
     &                ,ii),iseas)
                    else
                      stagedff=0.0
                    endif
                    diff_stage=stage_diff-stagedff
                    tot_pump=0.0
                    dschgdn=strout(cnlstg(l29c,ii,1)-diff_stage,
     &               idncnl(l29c,ii),idncnl(l29c,ii),kk,0,crmin,
     &               volwtr,prevqu,rain,iter,tot_pump,jday,
     &               tempf_volwtr,tempf_weircff,0)
                    dischgt=dischgt+dschgdn
c                    if (conv.eq.1.) print *,dischgt,DSCHGDN,' KKKK '
c    $                ,flnm(istr(idncnl(l29c,ii),kk))
                  END DO
                  avvol_dns334=dischgt-qu(idncnl(l29c,ii))/86400.-
     &             tovflc(idncnl(l29c,ii),cnlstg(l29c,ii,1)-
     &             stage_diff,0,0.,iter,2)/86400.-cgseep(idncnl(l29c,
     &             ii),cnlstg(l29c,ii,1)-stage_diff,0,0.)/86400.+
     &             (cnlstg(l29c,ii,1)-stage_diff-chdep(idncnl(l29c,ii)
     &             ))*cl(idncnl(l29c,ii))*width(idncnl(l29c,ii))/
     &             86400.-dischg_from_l30
c                 up_inflow_dn_cnl=qu(idncnl(l29c,ii))/86400.
                  avvol_dns334=amax1(avvol_dns334,0.0)
c                  if (conv.eq.1.)
c    $            print *,avvol_dns334,capac_s334,dischgt,' AVVOLDNS '
c    $             ,CNLSTG(l29c,ii,1),qu(idncnl(l29c,ii))/86400.
c    $             ,dischg_from_l30,frac_struc_cap(l29c,ii,month)
c    $             ,tovflc(idncnl(l29c,ii),CNLSTG(l29c,ii,1)
c    $             - stage_diff,0,0.,ITER,2)/86400.
c    $             ,cgseep(idncnl(l29c,ii),CNLSTG(l29c,ii,1)
c    $             - stage_diff,0,0.)/86400.,dischg_from_l30
                END IF
              END IF
            END DO
            reg_bot_s333=regstg_wca(iwca3a,
     &       ibot_zone_indx_s333reg(iwca3a+1))
            IF (trig_wca_stg(iwca3a)
     &       .gt.reg_bot_s333.or.nint(ws_need_l31nc_orig).eq.0) THEN
              IF (trig_wca_stg(iwca3a).lt.regstg_wca(iwca3a,
     &         nzone(iwca3a+1)-2)) THEN
                srs_targ_not_met=amax1(prev_totf-prev_total_flow_s12-
     &           s355prevout,0.0)
              ELSE
                srs_targ_not_met=9999999
              END IF
              IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
                avail_cap_s333=amin1(s333_max_cap,total_des_cap)-cout
              ELSE
                avail_cap_s333=amin1(s333_max_cap,total_des_cap)-cout
                avail_cap_s333=avail_cap_s333*frac_avail_cap_s333
              END IF
              avail_cap_s333=amin1(avail_cap_s333,srs_targ_not_met-
     &         s333ftargprev)
```

```
                avail_cap_s333=amax1(avail_cap_s333,0.0)
                IF (stage_g3273.gt.stg_trigger_fl(n,ind1,1)) THEN
                  add_s333_fc_s334=amin1(avail_cap_s333,capac_s334,
     &             avvol_dns334)
                ELSE
                  IF (is334_open_flg.eq.0) THEN
                    add_s333_fc_s334=avail_cap_s333
                  ELSE
                    add_s333_fc_s334=amin1(avail_cap_s333,capac_s334,
     &               avvol_dns334)
                  END IF
                END IF
c                if (conv.eq.1.) print *,add_s333_fc_s334
c    $              ,avail_cap_s333,s333_max_cap,total_des_cap
c    $              ,cout,capac_s334,avvol_dns334,'  S333FC   '
c    $              ,srs_targ_not_met,prev_TOTF,s333ftargprev
c    $              ,prev_total_flow_s12
                cout=add_s333_fc_s334+cout
                s333_reg_lec=add_s333_fc_s334
                IF (trig_wca_stg(iwca3a).gt.reg_bot_s333) THEN
                  s333_reg_lec_abve_sch=s333_reg_lec
                  s333_reg_lec_bel_sch=0.0
                ELSE
                  s333_reg_lec_abve_sch=0.0
                  s333_reg_lec_bel_sch=s333_reg_lec
                END IF
              END IF
            END IF
          END IF
C
          rmflws333_temp=rmflws333
          rem_wsneed_s151=tdqu_dn(l30)+tdqu_dn(icnl_dn_s31)-
     &     kflo(ifls151)
          rem_wsneed_s151=amax1(rem_wsneed_s151,0.0)
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
            rlocal_flow=0.0
            avvol_up=-901.
            IF (lp(23)) THEN
              WRITE (62,90) ifyr+iyear-1,month,iday,flnm(iflo),chdep(n),
     &         tw,s333mx,avvol_up,kflo(iflo)+rlocal_flow,s333_flow_targ
            END IF
          END IF
C
C-----------------------------------------------------------------
C                     S-343A AND B  (PADRICK PASS GATED CULVERTS)
C-----------------------------------------------------------------
```

```
C
      ELSE IF (flnm(iflo).eq.strname_spec(52)) THEN
        tw_node_stg=0.0
        iup_wca=idn_wca_indx(is8bsn-6)
        DO i=1,ncells_rec(n,ind1)
          gwdpth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
     &     s(nodrcv(n,ind1,i))
          IF (pond(nodrcv(n,ind1,i)).gt.gwdpth) THEN
            stage_loc=pond(nodrcv(n,ind1,i))-gwdpth+ells(nodrcv(n,
     &       ind1,i))
          ELSE
            stage_loc=(pond(nodrcv(n,ind1,i))-gwdpth)/s(nodrcv(n,ind1,
     &       i))+ells(nodrcv(n,ind1,i))
          END IF
          tw_node_stg=tw_node_stg+stage_loc/ncells_rec(n,ind1)
        END DO
        avvol_dn=0.0
        regt=regstg_wca(iup_wca,1)
        DO i=1,ncells_rec(n,ind1)
          avvol=max(chdep(n)-min(tw_node_stg,ells(nodrcv(n,ind1,i)
     &     )),0.0)*s(nodrcv(n,ind1,i))+max(chdep(n)-
     &     max(tw_node_stg,ells(nodrcv(n,ind1,i))),0.0)
          avvol_dn=avvol_dn+avvol*gdar/86400.
        END DO
        IF (month.eq.7.and.iday.gt.15) THEN
          frac_s343ab=frac_capac_s343ab(month+1)
        ELSE
          frac_s343ab=frac_capac_s343ab(month)
        END IF
        capac_struc=frac_s343ab*discoef(n,ind1,1,1)
        tw_for_constrnt=0.0
        DO k=1,no_cells_trig_s12s343(1)
          tw_for_constrnt=h(icell_loc_trig_s12s343(1))+
     &     pond(icell_loc_trig_s12s343(1))+tw_for_constrnt
        END DO
        tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(1)
        IF (tw_for_constrnt.gt.rmax_stg_s12s343(1,month)) THEN
          capac_struc=0.0
        END IF
        IF ((trig_wca_stg(iup_wca).gt.regt.and.bcnpenv.eq.'FALSE')
     &   .or.(lwslo(ibcnp).and.bcnpenv.eq.'TRUE ')) THEN
          cout=min(capac_struc,avvol_dn)
        ELSE
          cout=0.0
        END IF
        IF (conv.eq.1) THEN
          IF (irceive(n,ind1).eq.2) THEN
            DO i=1,ncells_rec(n,ind1)
              pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &         86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &         (nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
            END DO
          ELSE
            qu(idncl)=qu(idncl)+cout*86400.
          END IF
          kflo(iflo)=cout
```

```
          END IF
C-----------------------------------------------------------------
C                      S-344  (L28 CROSS OVER CULVERTS)
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(53)) THEN
          tw_node_stg=0.0
          iup_wca=idn_wca_indx(is8bsn-6)
          DO i=1,ncells_rec(n,ind1)
            gwdpth=(ells(nodrcv(n,ind1,i))-h(nodrcv(n,ind1,i)))*
     &       s(nodrcv(n,ind1,i))
            IF (pond(nodrcv(n,ind1,i)).gt.gwdpth) THEN
              stage_loc=pond(nodrcv(n,ind1,i))-gwdpth+ells(nodrcv(n,
     &         ind1,i))
            ELSE
              stage_loc=(pond(nodrcv(n,ind1,i))-gwdpth)/s(nodrcv(n,ind1,
     &         i))+ells(nodrcv(n,ind1,i))
            END IF
            tw_node_stg=tw_node_stg+stage_loc/ncells_rec(n,ind1)
          END DO
C
          regt=regstg_wca(iup_wca,1)
          avvol_dn=0.0
          DO i=1,ncells_rec(n,ind1)
            avvol=max(min(chdep(n),ells(nodrcv(n,ind1,i)))-
     &       min(tw_node_stg,ells(nodrcv(n,ind1,i))),0.0)*s(nodrcv(n,
     &       ind1,i))+max(chdep(n)-max(tw_node_stg,ells(nodrcv(n,
     &       ind1,i))),0.0)
            avvol_dn=avvol_dn+avvol*gdar/86400.
          END DO
          IF (month.eq.7.and.iday.gt.15) THEN
            frac_s343ab=frac_capac_s343ab(month+1)
          ELSE
            frac_s343ab=frac_capac_s343ab(month)
          END IF
          capac_struc=frac_s343ab*discoef(n,ind1,1,1)
          tw_for_constrnt=0.0
          DO k=1,no_cells_trig_s12s343(1)
            tw_for_constrnt=h(icell_loc_trig_s12s343(1))+
     &       pond(icell_loc_trig_s12s343(1))+tw_for_constrnt
          END DO
          tw_for_constrnt=tw_for_constrnt/no_cells_trig_s12s343(1)
          IF (tw_for_constrnt.gt.rmax_stg_s12s343(1,month)) THEN
            capac_struc=0.0
          END IF
          IF ((trig_wca_stg(iup_wca).gt.regt.and.bcnpenv.eq.'FALSE')
     &     .or.(lwslo(ibcnp).and.bcnpenv.eq.'TRUE ')) THEN
            cout=min(capac_struc,avvol_dn)
          ELSE
            cout=0.0
          END IF
C          if (conv.eq.1.)
C    $     print *,tw_for_constrnt,rmax_stg_s12s343(1,month),frac_s343ab
C    $      ,capac_struc,trig_wca_stg(iup_wca),REGT,avvol_dn,cout
C    $      ,chdep(n),tw_node_stg,ells(NODRCV(n,ind1,1))
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.2) THEN
```

```
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            ELSE
              qu(idncl)=qu(idncl)+cout*86400.
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------------------
C             WATER SUPPLY FROM EASTERN M-CANAL TO CLEAR LAKE
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(54)) THEN
          IF (chdep(n).gt.chdep(idncl)) THEN
            capacity=discoef(n,ind1,1,1)*(chdep(n)-chdep(idncl))**pwr(n,
     &       ind1,1,1)
          ELSE
            capacity=0.0
          END IF
          IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
            capac=5.0*(chdep(n)-rmnstgs(n,ind1,iseas))*capacity
            capac=amin1(capacity,capac)+qu(n)/86400.
            avvol_fc=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &       cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,
     &       ind1,iseas),0,0.,iter,1)+qu(n)
            avvolfc=amax1(avvol_fc,0.0)/86400.
            coutws=amin1(qu(n)/86400.,dqu(ilkmg))
            cout=amax1(avvolfc,coutws)
            cout=amin1(cout,capac)
          ELSE
            avvol_up=qu(n)
            avvol_up=max(avvol_up/86400.,0.0)
            cout=min(avvol_up,dqu(ilkmg),capacity)
          END IF
          IF (conv.eq.1) THEN
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            qu(idncl)=qu(idncl)+cout*86400.
            dqu(ilkmg)=dqu(ilkmg)-cout
            kflo(iflo)=cout
          END IF
C
C
C-----------------------------------------------------------------
C                S-319 PROPOSED PUMP FOR C51W
C-----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(55)) THEN
          IF (chdep(n).gt.chdep(ic51)+hdc(ic51,iseas)) THEN
            capac_s155a=625.0*(chdep(n)-(chdep(ic51)+hdc(ic51,iseas)))**
     &       0.5
          ELSE
            capac_s155a=0.0
          END IF
          CALL match (name_res_spec_code(6), 1, 6, resname, ntotres+1,
     &     6, ires_no_sta1e)
          cout=amin1(capac_s155a,out_flow_east)
```

```
          capac_s155a=capac_s155a-cout
c
c  route L8 basin runoff thru C51W/S-155A to east if
c  iroute_l8rnf_to_sta_opt = 0,otherwise route L8 runoff
c  to STA-1E.
c
          if (iroute_l8rnf_to_sta_opt .eq. 0) then
             cout=amin1(capac_s155a,flfrml8toc51w)+cout
             out_flow_east=out_flow_east+flfrml8toc51w
             capac_s155a=capac_s155a-cout
          endif
c
          cout_to_c51_frm_l8bprs_1=amin1(cout_to_c51_frm_l8bprs,
     &     capac_s155a)
          out_flow_east=out_flow_east+cout_to_c51_frm_l8bprs_1
          cout=cout+cout_to_c51_frm_l8bprs_1
          capac_s155a=capac_s155a-cout_to_c51_frm_l8bprs_1
          avvol_up=(chdep(n)-crmin(n,iseas))*cl(n)*width(n)+cgseep(n,
     &     crmin(n,iseas),0,0.)+tovflc(n,crmin(n,iseas),0,0.,iter,1)+
     &     qu(n)-cout*86400.- tot_avl_l8_bpit_res*86400.
          avvol_up=max(avvol_up/86400.,0.0)
          tdqu=amax1(dqu(ic51)-out_flow_east,0.0)
          wsdel=min(avvol_up,tdqu)
          IF (chdep(n).gt.chdep(ic51)+hdc(ic51,iseas)) THEN
            cout=cout+amin1(wsdel,capac_s155a)
          END IF
          out_flow_east=out_flow_east+amin1(wsdel,capac_s155a)
          avvpmp=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)-out_flow_east*86400.
     &     - tot_avl_l8_bpit_res*86400.
          avvpmp=amax1(avvpmp/86400.,0.0)
          cout=amin1(avvpmp,discoef(n,ind1,1,1))+cout
          flwexc=amax1(avvpmp-discoef(n,ind1,1,1),0.0)
          IF (chdep(n).gt.chdep(ic51)+hdc(ic51,iseas)) THEN
            cout=cout+amin1(flwexc,capac_s155a)
            out_flow_east=out_flow_east+amin1(flwexc,capac_s155a)
            capac_s155a=capac_s155a-amin1(flwexc,capac_s155a)
          END IF
          IF (conv.eq.1) THEN
            CALL match (name_res_spec_code(6), 1, 6, resname, ntotres,
     &       6, ista1ew_res)
            CALL match (name_res_spec_code(21), 1, 6, resname, ntotres,
     &       6, ista1e_res)
            if (irceive(n,ind1).eq.2) THEN
             sum_res_area_sta1e = 0.0
             if (ista1ew_res .ne. 0) then
                sum_res_area_sta1e =sum_res_area_sta1e
     &          + res_area(ista1ew_res)
             endif
             if (ista1e_res .ne. 0) then
                sum_res_area_sta1e =sum_res_area_sta1e
     &          + res_area(ista1e_res)
             endif
             if (ista1ew_res .ne. 0) then
                frac_s319_sta1ew = res_area(ista1ew_res)
     &           / sum_res_area_sta1e
```

```
            else
               if (ista1e_res .eq. 0) then
                 frac_s319_sta1ew = 1.0 / ncells_rec(n,ind1)
               endif
            endif
            if (ista1e_res .ne. 0) then
               frac_s319_sta1ee = res_area(ista1e_res)
     &         / sum_res_area_sta1e
            else
               if (ista1ew_res .eq. 0) then
                 frac_s319_sta1ee = 1.0 / ncells_rec(n,ind1)
               endif
            endif
           endif
           kflo(iflo)=amin1(avvpmp,discoef(n,ind1,1,1))
           IF (irceive(n,ind1).eq.2) THEN
             nodrcv_w = 31 - minx(53) + 1 + isum(53)
             nodrcv_e = 32 - minx(53) + 1 + isum(53)
             DO i=1,ncells_rec(n,ind1)
              if ( nodrcv(n,ind1,i).eq.nodrcv_w) then
               pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+kflo(iflo)
     &         *86400.*frac_s319_sta1ew/(gdar*(1.0
     &         -carfact(nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
              elseif (nodrcv(n,ind1,i).eq.nodrcv_e) then
               pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+kflo(iflo)
     &         *86400.*frac_s319_sta1ee/(gdar*(1.0
     &         -carfact(nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
              else
               pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+kflo(iflo)
     &         *86400./(gdar*ncells_rec(n,ind1)*(1.0
     &         -carfact(nodrcv(n,ind1,i))-total_small_afact
     &         (nodrcv(n,ind1,i))))
              endif
             END DO
           ELSE
             qu(idncnl(n,ind1))=qu(idncnl(n,ind1))+kflo(iflo)
     &        *86400.
           END IF
           if (ista1ew_res.ne.0) res_struc_inflw(ista1ew_res)
     &      = res_struc_inflw(ista1ew_res)+kflo(iflo)*1.9835
     &      * frac_s319_sta1ew
           IF (ista1e_res.ne.0) res_struc_inflw(ista1e_res)=
     &      res_struc_inflw(ista1e_res)+kflo(iflo)*1.9835
     &      * frac_s319_sta1ee
           flws319=kflo(iflo)
           IF (chdep(ic51w).gt.chdep(ic51)+hdc(ic51,iseas)) THEN
             qu(ic51)=qu(ic51)+(cout-flws319)*86400.
             rfls155a=(cout-flws319)+rfls155a
             dqu(ic51)=dqu(ic51)-amin1(dqu(ic51),rfls155a)
           END IF
          END IF
C
C-----------------------------------------------------------------
C                        G-72
C-----------------------------------------------------------------
```

```
C
        ELSE IF (flnm(iflo).eq.strname_spec(56)) THEN
          IF (dqu(idncl).gt.0.0) THEN
            IF (nprty(n,ind1).ne.0) THEN
              cout=amin1(dqu(idncl),tempvol)
            ELSE
              volrat=tempvol/tdmnd
              wratio(n)=amin1(volrat,1.0)
              wsneed=dqu(idncl)*wratio(n)
              cout=wsneed
              tdmnd=tdmnd-dqu(idncl)
            END IF
          ELSE
            cout=0.0
          END IF
          tempvol=tempvol-cout
C
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            dqu(idncl)=dqu(idncl)-cout
            kflo(iflo)=cout
          END IF
C-------------------------------------------------------
C                 WSTC8
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(57)) THEN
          IF (dqu(idncl).gt.0.0) THEN
            IF (nprty(n,ind1).ne.0) THEN
              cout=amin1(dqu(idncl),tempvol)
            ELSE
              volrat=tempvol/tdmnd
              wratio(n)=amin1(volrat,1.0)
              wsneed=dqu(idncl)*wratio(n)
              cout=wsneed
              tdmnd=tdmnd-dqu(idncl)
            END IF
          ELSE
            cout=0.0
          END IF
          tempvol=tempvol-cout
C
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            dqu(idncl)=dqu(idncl)-cout
            kflo(iflo)=cout
          END IF
        ELSE IF (flnm(iflo).eq.strname_spec(70)) THEN
          IF (nint(crel(idncl,1)).gt.0) THEN
            IF (chdep(idncl).le.crel(idncl,1)) THEN
              IF (chdep(n).gt.rmnstgs(n,ind1,iseas)+0.5) THEN
                cout=capacws(imgt)
              ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
                cout=2.0*capacws(imgt)*(chdep(n)-rmnstgs(n,ind1,iseas))
              ELSE
                cout=0.0
```

```
              END IF
            ELSE
              cout=0.0
            END IF
          ELSE
            IF (chdep(n).gt.rmnstgs(n,ind1,iseas)+0.5) THEN
              cout=capacws(imgt)
            ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
              cout=2.0*capacws(imgt)*(chdep(n)-rmnstgs(n,ind1,iseas))
            ELSE
              cout=0.0
            END IF
          END IF
          stage_m=hw-stgdiff_canal_tw
          IF (nint(crel(idncl,1)).gt.0) THEN
            IF (stage_m.lt.crel(idncl,1)) THEN
              IF (stage_m.gt.rmnstgs(idncl,1,iseas)+0.5) THEN
                cout_dwn=strdcap(idncl,1,1,imgt)
              ELSE IF (stage_m.gt.rmnstgs(idncl,1,iseas)) THEN
                cout_dwn=2.0*strdcap(idncl,1,1,imgt)*(stage_m-
     &           rmnstgs(idncl,1,iseas))
              ELSE
                cout_dwn=0.0
              END IF
            ELSE
              cout_dwn=0.0
            END IF
          ELSE
            IF (stage_m.gt.rmnstgs(idncl,1,iseas)+0.5) THEN
              cout_dwn=strdcap(idncl,2,1,imgt)
            ELSE IF (stage_m.gt.rmnstgs(idncl,1,iseas)) THEN
              cout_dwn=2.0*strdcap(idncl,1,1,imgt)*(stage_m-
     &         rmnstgs(idncl,1,iseas))
            ELSE
              cout_dwn=0.0
            END IF
          END IF
C
          vol_lim=(hw-tw)*cl(idncl)*width(idncl)-cgseep(idncl,stage_m,0,
     &     0.)-tovflc(idncl,stage_m,0,0.,iter,2)+cout_dwn*86400.
          vol_lim=amax1(vol_lim,0.0)/86400.
          cout=amin1(cout,strdcap(n,ind1,1,imgt),vol_lim)
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            outflow_to_loxrv=cout
          END IF
C-----------------------------------------------
C           WS TO LOX RIVER FROM UPSTREAM
C-----------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(71)) THEN
          avvolup=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvolup=amax1(avvolup/86400.,0.0)
          cout=amin1(avvolup,discoef(n,ind1,1,imgt))
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
```

```
            kflo(iflo)=cout
          END IF
C-------------------------------------------
C            G92TRV
C-------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(72)) THEN
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up/86400.,0.0)
          IF (tdmnd.gt.0.0) THEN
            ratio_ws=amin1(tempvol/tdmnd,1.0)
          ELSE
            ratio_ws=0.0
          END IF
          IF (nint(crel(idncl,1)).gt.0) THEN
            IF (chdep(idncl).lt.crel(idncl,1)) THEN
              twmax=hw-0.20
              twmax=min(twmax,crel(idncl,1))
              frac2=max(max(twmax,13.0)-rmnstgs(n,ind1,iseas),0.0)/
     &         (crel(n,1)-rmnstgs(n,ind1,iseas))
              frac2=min(frac2,1.0)
              IF (twmax.gt.rmnstgs(idncl,2,iseas)) THEN
                frac_flow=max(twmax-chdep(idncl),0.0)/(twmax-
     &           rmnstgs(idncl,2,iseas))
              ELSE
                frac_flow=0.0
              END IF
              add_flow=min(frac_flow,1.0)*strdcap(n,ind1,1,1)*frac2
              avvoldn=(twmax-tw)*cl(idncl)*width(idncl)-cgseep(idncl,
     &         twmax,0,0.)-tovflc(idncl,twmax,0,0.,iter,2)-qu(idncl)+
     &         add_flow*86400.
              avvoldn=max(avvoldn,0.0)/86400.
              IF (hw.gt.rmnstgs(n,ind1,iseas)+0.5) THEN
                cout=capacws(imgt)
              ELSE IF (hw.gt.rmnstgs(n,ind1,iseas)) THEN
                cout=2.0*capacws(imgt)*(hw-rmnstgs(n,ind1,iseas))
              ELSE
                cout=0.0
              END IF
              cout=min(cout,avvoldn)
            ELSE
              cout=0.0
            END IF
          ELSE
            IF (chdep(idncl).lt.rmxstg(idncl,1,iseas)) THEN
              twmax=hw-0.20
              twmax=min(twmax,rmxstg(idncl,2,iseas))
              frac2=max(twmax-rmnstgs(n,ind1,iseas),0.0)/(crel(n,1)-
     &         rmnstgs(n,ind1,iseas))
              frac_flow=max(twmax-chdep(idncl),0.0)/(twmax-
     &         rmnstgs(idncl,2,iseas))
              add_flow=min(frac_flow,1.0)*strdcap(n,ind1,1,1)*frac2
              avvoldn=(twmax-tw)*cl(idncl)*width(idncl)-cgseep(idncl,
     &         twmax,0,0.)-tovflc(idncl,twmax,0,0.,iter,2)-qu(n)+
     &         add_flow*86400.
              avvoldn=max(avvoldn,0.0)/86400.
```

```
              IF (hw.gt.rmnstgs(n,ind1,iseas)+0.5) THEN
                cout=capacws(imgt)
              ELSE IF (hw.gt.rmnstgs(n,ind1,iseas)) THEN
                cout=2.0*capacws(imgt)*(hw-rmnstgs(n,ind1,iseas))
              ELSE
                cout=0.0
              END IF
              cout=min(cout,avvoldn)
            ELSE
              cout=0.0
            END IF
          END IF
          cout=amin1(cout,strdcap(n,ind1,1,1))*ratio_ws
          tempvol=tempvol-amin1(tempvol,cout)
          tdmnd=tdmnd-strdcap(n,ind1,1,1)
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
          END IF
C-----------------------------------------------------
C             ITWCD LOWER BASIN OUTLET
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(73)) THEN
          capac=discoef(n,ind1,1,imgt)
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up/86400.,0.0)
C
          dischg=0.0
          DO is=1,nstrctr(idncl)
            dischg=strout(chdep(n),idncl,idncl,is,0,crmin,volwtr,prevqu,
     &       rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)+
     &       dischg
          END DO
          avvol_dn=(chdep(n)-tw)*cl(idncl)*width(idncl)-cgseep(idncl,
     &     chdep(n),0,0.)-tovflc(idncl,chdep(n),0,0.,iter,2)
     &     +dischg*86400.
          avvol_dn=max(avvol_dn,0.0)/86400.
          IF (tw.gt.rmnstgs(idncl,1,iseas)) THEN
            outflw_ds_cnl=discoef(idncl,1,1,imgt)*(tw-rmnstgs(idncl,1,
     &       iseas))**pwr(idncl,1,1,imgt)
          ELSE
            outflw_ds_cnl=0.0
          END IF
          avail_vol_ds=strdcap(idncl,1,1,imgt)-outflw_ds_cnl
          avail_vol_ds=amax1(avail_vol_ds,0.0)
          cout=min(avvol_up,avail_vol_ds,capac,avvol_dn)
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            kflo(iflo)=cout
            outflw_from_milb=cout
            pot_out_from_itlb=avail_vol_ds
          END IF
C-----------------------------------------------------
C             ITWCD UPPER BASIN OUTLET
C-----------------------------------------------------
```

```
         ELSE IF (flnm(iflo).eq.strname_spec(74)) THEN
           IF (hw.gt.tw) THEN
             capac=discoef(n,ind1,1,imgt)*(hw-tw)**pwr(n,ind1,1,imgt)
           ELSE
             capac=0.0
           END IF
           avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &      cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &      iseas),0,0.,iter,1)+qu(n)
           avvol_up=max(avvol_up/86400.,0.0)
C
           IF (pot_out_from_itlb.ge.discoef(idncl,1,1,imgt)) THEN
             poten_flow=amax1(discoef(idncl,1,1,imgt)-outflw_from_milb,
     &        0.0)
             cout=min(poten_flow,avvol_up,capac)
           ELSE
             cout=0.0
           END IF
           IF (conv.eq.1.) THEN
             qu(idncl)=qu(idncl)+cout*86400.
             IF (idncl.lt.n) THEN
               rloc_excss_to_canl_next_step(idncl)=
     &          rloc_excss_to_canl_next_step(idncl)+cout*86400.
             END IF
             upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
             kflo(iflo)=cout
           END IF
C-------------------------------------------------------------
C          PROPSED STRUC IN L-37 Canal to PASS ENVIRONMENTAL FLOWS
C-------------------------------------------------------------
         ELSE IF (flnm(iflo).eq.strname_spec(59).or.flnm(iflo)
     &   .eq.strname_spec(60)) THEN
           IF (chdep(n).gt.rmxstg(n,ind1,iseas)) THEN
             coutfl=capacfl(imgt,1)
           ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
             pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**3*
     &        (chdep(n)-rmnstgs(n,ind1,iseas))**3
             coutfl=pctgo*capacfl(imgt,1)
           ELSE
             coutfl=0.0
           END IF
           IF (lwslo(ig1502)) THEN
             cout=amax1(coutfl,qu(n)/86400.)
             cout=amin1(cout,capacfl(imgt,1))
           ELSE
             cout=coutfl
           END IF
           IF (conv.eq.1.) THEN
             IF (irceive(n,ind1).eq.1) THEN
               qu(idncl)=qu(idncl)+cout*86400.
               IF (idncl.lt.n) THEN
                 rloc_excss_to_canl_next_step(idncl)=
     &            rloc_excss_to_canl_next_step(idncl)+cout*86400.
               END IF
               upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
             ELSE
               DO i=1,ncells_rec(n,ind1)
```

```
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------------
C          PROPSED STRUC IN L-30 to PASS ENVIRONMENTAL FLOWS
C-----------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(61)) THEN
          IF (chdep(n).gt.rmxstg(n,ind1,iseas)) THEN
            coutfl=capacfl(imgt,1)
          ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
            pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**3*
     &       (chdep(n)-rmnstgs(n,ind1,iseas))**3
            coutfl=pctgo*capacfl(imgt,1)
          ELSE
            coutfl=0.0
          END IF
          IF (lwslo(ig1502)) THEN
            cout=amax1(coutfl,qu(n)/86400.)
            cout=amin1(cout,capacfl(imgt,1))
          ELSE
            cout=coutfl
          END IF
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
              dqu(idncl)=dqu(idncl)-amin1(cout,dqu(idncl))
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------------
C          PROP PUMPAGE OF C11W BASIN RUNOFF FROM NLKBLT TO C9RES
C-----------------------------------------------------------
        ELSE IF (flnm(iflo).eq.'NLBC9R') THEN
          avail_cap=amax1(capacfl(imgt,1)-tot_flw_to_c9r_frm_nlkblt,0.0)
          IF (nint(flow_to_lbn_frm_c11).gt.0) THEN
            cout=amin1(avail_cap,volwtr(irsrcvr(n,ind1,1)),
     &       flow_to_lbn_frm_c11)
          ELSE
            cout=0.0
          END IF
```

```
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.2) THEN
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            ELSE
              qu(idncl)=qu(idncl)+cout*86400.
            END IF
            tot_flw_to_c9r_frm_nlkblt=tot_flw_to_c9r_frm_nlkblt+cout
C         print *,cout,avail_cap,capacfl(imgt,1),flow_to_lbn_frm_c11
C    $      ,tot_flw_to_c9r_frm_nlkblt,volwtr(IRSRCVR(N,IND1,1))
C    $      ,resname(IRSRCVR(N,IND1,1))
          END IF
C
C-------------------------------------------------------------
C          OUTFLOW FRM NLKBLT TO C9RES TO KEEP C9RES WET
C-------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.'NLC9MN') THEN
          avail_vol=(chdep(n)-stgintke(n,ind1,1,imgt))*cl(n)*width(n)+
     &     cgseep(n,stgintke(n,ind1,ns,imgt),0,0.)+tovflc(n,stgintke(n,
     &     ind1,ns,imgt),0,0.,iter,1)+qu(n)
          avail_vol=amax1(avail_vol,0.0)/86400.
          avail_cap=amax1(capacfl(imgt,1)-tot_flw_to_c9r_frm_nlkblt,0.0)
          dpthgw=(ells(nodrcv(n,ind1,1))-h(nodrcv(n,ind1,1)))*
     &     s(nodrcv(n,ind1,1))
          IF (pond(nodrcv(n,ind1,1)).gt.dpthgw) THEN
            c9r_stage=pond(nodrcv(n,ind1,1))-dpthgw+ells(nodrcv(n,ind1,
     &       1))
          ELSE
            c9r_stage=(pond(nodrcv(n,ind1,1))-dpthgw)/s(nodrcv(n,ind1,1)
     &       )+ells(nodrcv(n,ind1,1))
          END IF
          IF (c9r_stage.le.ells(nodrcv(n,ind1,1)).or.lwsdel) THEN
            lwsdel=.true.
            sum_c9rrain=sum_c9rrain+rain(nodrcv(n,ind1,1))/12.
            sum_c9ret=sum_c9ret+daily_et(nodrcv(n,ind1,1))
            rfetnet=amax1(sum_c9rrain-sum_c9ret,0.0)
            etrfnet=amax1(sum_c9ret-sum_c9rrain,0.0)
            lwsdel=.not.(c9r_stage.gt.ells(nodrcv(n,ind1,1))+
     &       amin1(rfetnet-etrfnet*s(nodrcv(n,ind1,1)),0.5)+amax1((0.5-
     &       (rfetnet-etrfnet*s(nodrcv(n,ind1,1))))*(860./(gdar/43560.))
     &       ,0.0))
            IF (lwsdel) THEN
              cout_for_ws=(0.5-(rfetnet-etrfnet*s(nodrcv(n,ind1,1))))*
     &         860./1.9835
            ELSE
              sum_c9rrain=0.0
              sum_c9ret=0.0
              cout_for_ws=0.0
            END IF
            cout=amin1(avail_vol,avail_cap,cout_for_ws)
          ELSE
            lwsdel=.false.
            cout=0.0
```

```
            sum_c9rrain=0.0
            sum_c9ret=0.0
          END IF
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.2) THEN
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            ELSE
              qu(idncl)=qu(idncl)+cout*86400.
            END IF
C           print *,c9r_stage,ELLS(NODRCV(n,ind1,1)),LWSDEL,cout_for_ws
C    $             ,cout,avail_vol,avail_cap,capacfl(imgt,1)
C    $              ,tot_flw_to_c9r_frm_nlkblt,STGINTKE(N,IND1,1,IMGT)
C    $              ,ibsn(NODRCV(n,ind1,1))
            tot_flw_to_c9r_frm_nlkblt=tot_flw_to_c9r_frm_nlkblt+cout
          END IF
C-------------------------------------------------
C              S-335 IOP
C-------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(96)) THEN
          IF (chdep(n).gt.rmxstg(n,ind1,iseas)) THEN
            coutfl=capacfl(imgt,1)
          ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
            pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**3*
     &       (chdep(n)-rmnstgs(n,ind1,iseas))**3
            coutfl=pctgo*capacfl(imgt,1)
          ELSE
            coutfl=0.0
          END IF
          CALL match ('L30  ', 1, 5, rlist_lvname, nlevs, 5, ilv30)
          IF (nint(s337fc) .gt. 0) THEN
            struc_inflw=amax1(qu(n)/86400.-tseepge(ilv30),0.0)
            cout = amin1(coutfl,capacfl(imgt,1),struc_inflw)
          ELSE
            cout = 0.0
          ENDIF
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
```

```
          ENDIF

C-----------------------------------------------------
C              S-335
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(6)) THEN
          IF (chdep(n).gt.rmxstg(n,ind1,iseas)) THEN
            coutfl=capacfl(imgt,1)
          ELSE IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
            pctgo=1.0/(rmxstg(n,ind1,iseas)-rmnstgs(n,ind1,iseas))**3*
     &       (chdep(n)-rmnstgs(n,ind1,iseas))**3
            coutfl=pctgo*capacfl(imgt,1)
          ELSE
            coutfl=0.0
          END IF
          IF (lwslo(ig1502)) THEN
            CALL match ('L30  ', 1, 5, rlist_lvname, nlevs, 5, ilv30)
            struc_inflw=amax1(qu(n)/86400.-tseepge(ilv30),0.0)
            cout=amax1(coutfl,struc_inflw)
            cout=amin1(cout,capacfl(imgt,1))
          ELSE
            cout=coutfl
          END IF
          IF (conv.eq.1.) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              IF (idncl.lt.n) THEN
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-------------------------------------------------------------
C                     C-44 CANAL
C-------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(62)) THEN
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up/86400.,0.0)
          cout=avvol_up
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            kflo(iflo)=cout
          END IF
C
C-------------------------------------------------------------
```

```
C                        S-30
C-------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(63)) THEN
          IF (dqu(idncl).gt.0.0) THEN
            IF (nprty(n,ind1).ne.0) THEN
              cout=amin1(dqu(idncl),tempvol)
            ELSE
              volrat=tempvol/tdmnd
              wratio(n)=amin1(volrat,1.0)
              wsneed=dqu(idncl)*wratio(n)
              cout=wsneed
              tdmnd=tdmnd-dqu(idncl)
            END IF
          ELSE
            cout=0.0
          END IF
          tempvol=tempvol-cout
C
          IF (conv.eq.1.) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            dqu(idncl)=dqu(idncl)-cout
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------
C                 S-174 or S332D
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(64)) THEN
          deltah=chdep(n)-(chdep(idncl)+hdc(idncl,iseas))
          IF (month.eq.7.and.iday.gt.15) THEN
            frac_struc_cap_s332d=frac_struc_cap(n,ind1,month+1)
          ELSE
            frac_struc_cap_s332d=frac_struc_cap(n,ind1,month)
          END IF
          IF (irating(n,ind1,1,imgt).ne.2) THEN
            IF (deltah.gt.rminhd(n,ind1)) THEN
              s174cap=discoef(n,ind1,1,imgt)*(deltah-rminhd(n,ind1))**
     &         pwr(n,ind1,1,imgt)*frac_struc_cap_s332d
            ELSE
              s174cap=0.0
            END IF
          ELSE
            s174cap=discoef(n,ind1,1,imgt)*frac_struc_cap_s332d
          END IF
          s174cap=amin1(s174cap,strdcap(n,ind1,1,imgt))
          IF (opt_flow_to_ts.eq.'TSRFPLN') THEN
            avvolq_exc=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &       cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,
     &       ind1,iseas),0,0.,iter,1)+qu(n)
            avvolq_exc=amax1(avvolq_exc,0.0)/86400.
            IF (nint(tdmnd_ds).gt.0) THEN
              tdmnd_ds1=amax1(tdmnd_ds-dqu(idncl),0.0)
              avvolq_exc=amax1(avvolq_exc-tdmnd_ds1,0.0)
            END IF
            IF (nint(dqu(idncl)).eq.0) THEN
              avvolq=avvolq_exc
```

```
            ELSE
              avvolq=(chdep(n)-stgintke(n,ind1,1,imgt))*cl(n)*width(n)+
     &         cgseep(n,stgintke(n,ind1,1,imgt),0,0.)+tovflc(n,
     &         stgintke(n,ind1,1,imgt),0,0.,iter,1)+qu(n)
              avvolq=max(avvolq,0.0)/86400.
            END IF
            IF (dqu(idncl).gt.0.0) THEN
              IF (tdmnd.gt.0.0) THEN
                ratio=tempvol/tdmnd
              ELSE
                ratio=0.0
              END IF
              wratio(n)=amin1(ratio,1.0)
              IF (nprty(n,ind1).eq.0) THEN
                coutws=min(s174cap,wratio(n)*dqu(idncl),avvolq)
                tdmnd=tdmnd-dqu(idncl)
              ELSE
                coutws=min(s174cap,dqu(idncl),tempvol,avvolq)
              END IF
            ELSE
              coutws=0.0
            END IF

            IF (dqu(idncl).gt.0.0) THEN
              cout=amax1(coutws,avvolq_exc)
              cout=amin1(cout,s174cap)
            ELSE
              IF (main_pres_level_fl_prot_ts.eq.'TRUE ') THEN
                cout=min(s174cap,avvolq)
              ELSE
                cout=0.0
              END IF
            END IF
          ELSE
            cout=0.0
          END IF
          tempvol=tempvol-cout
          IF (conv.eq.1.) THEN
C        print *,S174CAP,DQU(IDNCL),tdmnd_ds,dqu(ic111),COUT,'  S174'
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
              upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
              dqu(idncl)=dqu(idncl)-amin1(cout,dqu(idncl))
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-------------------------------------------------------
C                 S-175
C-------------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(65)) THEN
```

```
      deltah=chdep(n)-(chdep(idncl)+hdc(idncl,iseas))
      IF (deltah.gt.rminhd(n,ind1)) THEN
        s175cap=discoef(n,ind1,1,imgt)*(deltah-rminhd(n,ind1))**
&        pwr(n,ind1,1,imgt)
      ELSE
        s175cap=0.0
      END IF
      ls332mn=(month.ge.3.and.month.le.6)
&      .or.(month.eq.7.and.iday.le.15)
      IF (opt_flow_to_ts.eq.'TSRFPLN') THEN
        IF (main_pres_level_fl_prot_ts.eq.'FALSE') THEN
          IF (.not.lwslo(itylsl)) THEN
            stg_min=stage_import_wca(il31w_min_rf)+0.1
            stg_max=stage_import_wca(il31w_min_rf)+0.3
          ELSE
            stg_min=stage_import_wca(il31w_min_rf)+0.3
            stg_max=stage_import_wca(il31w_min_rf)+0.7
          END IF
        ELSE
          stg_max=rmxstg(n,ind1,iseas)
          stg_min=rmnstgs(n,ind1,iseas)
          IF (ls332mn.and.s332sparrow_opt.eq.'TRUE ') THEN
            stg_max=rmxstg(n,ind1,iseas)+1.0
            stg_min=rmnstgs(n,ind1,iseas)+1.0
          END IF
        END IF
        IF (chdep(n).gt.stg_max) THEN
          cout=s175cap
        ELSE IF (chdep(n).gt.stg_min) THEN
          cout=1.0/(stg_max-stg_min)*s175cap*(chdep(n)-stg_min)
        ELSE
          cout=0.0
        END IF
      ELSE
        stg_max=rmxstg(n,ind1,iseas)
        stg_min=rmnstgs(n,ind1,iseas)
        IF (ls332mn.and.s332sparrow_opt.eq.'TRUE ') THEN
          stg_max=rmxstg(n,ind1,iseas)+1.0
          stg_min=rmnstgs(n,ind1,iseas)+1.0
        END IF
        IF (chdep(n).gt.stg_max) THEN
          cout=s175cap
        ELSE IF (chdep(n).gt.stg_min) THEN
          cout=1.0/(stg_max-stg_min)*s175cap*(chdep(n)-stg_min)
        ELSE
          cout=0.0
        END IF
      END IF
      IF (conv.eq.1) THEN
        IF (irceive(n,ind1).eq.1) THEN
          qu(idncl)=qu(idncl)+cout*86400.
        ELSE
          DO i=1,ncells_rec(n,ind1)
            pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
&            86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
&            (nodrcv(n,ind1,i))-total_small_afact
&            (nodrcv(n,ind1,i))))
```

```
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------
C               S-332
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(22)) THEN
          IF (s332sparrow_opt.eq.'TRUE ') THEN
            IF (month.eq.7.and.iday.gt.15) THEN
              frac_struc_cap_s332=frac_struc_cap(n,ind1,month+1)
            ELSE
              frac_struc_cap_s332=frac_struc_cap(n,ind1,month)
            END IF
          ELSE
            frac_struc_cap_s332=frac_struc_cap(n,ind1,month)
          END IF
          s332cap=discoef(n,ind1,1,1)*frac_struc_cap_s332
          IF (opt_flow_to_ts.eq.'TSRFPLN') THEN
            IF (main_pres_level_fl_prot_ts.eq.'TRUE ') THEN
              stg_min=min(stage_outflw_wca(il31w_min_rf)+
     &         offset_out(il31w_min_rf,1),rmxstg(n,ind1,iseas))
              stg_min=max(stg_min,crmin(n,iseas)+0.25)
            ELSE
              stg_min=stage_outflw_wca(il31w_min_rf)+
     &         offset_out(il31w_min_rf,1)
            END IF
            avvolq=(chdep(n)-stg_min)*cl(n)*width(n)+cgseep(n,stg_min,0,
     &       0.)+tovflc(n,stg_min,0,0.,iter,1)+qu(n)
            avvolq=max(avvolq,0.0)/86400.
            cout=min(s332cap,avvolq)
          ELSE
            cout=0.0
          END IF
          IF (conv.eq.1) THEN
            IF (irceive(n,ind1).eq.1) THEN
              qu(idncl)=qu(idncl)+cout*86400.
            ELSE
              DO i=1,ncells_rec(n,ind1)
                pond(nodrcv(n,ind1,i))=pond(nodrcv(n,ind1,i))+cout*
     &           86400./(gdar*ncells_rec(n,ind1)*(1.0-carfact
     &           (nodrcv(n,ind1,i))-total_small_afact
     &           (nodrcv(n,ind1,i))))
              END DO
            END IF
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------
C                 S-331 AND S173
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(67)) THEN
          dpthgw=(ells(nodangl)-h(nodangl))*s(nodangl)
          IF (pond(nodangl).gt.dpthgw) THEN
            stg_angels1=pond(nodangl)-dpthgw+ells(nodangl)
          ELSE
            stg_angels1=(pond(nodangl)-dpthgw)/s(nodangl)+ells(nodangl)
          END IF
```

```
      dpthgw=(ells(ig2517)-h(ig2517))*s(ig2517)
      IF (pond(ig2517).gt.dpthgw) THEN
        stg_angels2=pond(ig2517)-dpthgw+ells(ig2517)
      ELSE
        stg_angels2=(pond(ig2517)-dpthgw)/s(ig2517)+ells(ig2517)
      END IF
      stg_angels=(stg_angels1+stg_angels2)/2.0
      langel=stg_angels.ge.stg_trigger_fl(n,ind1,3)
      IF (chdep(n).gt.chdep(idncl)) THEN
        str_cap(1)=discoef(n,ind1,1,1)*(chdep(n)-chdep(idncl))**
     &   pwr(n,ind1,1,1)
      ELSE
        str_cap(1)=0.0
      END IF
C
      langel_mx=.not.(stg_angels.lt.stg_trigger_fl(n,ind1,2))
      IF (stg_angels.gt.stg_trigger_fl(n,ind1,1).or.langel_mx) THEN
        langel_mx=.true.
        stg_min=stg_min_fc(n,ind1,2)
      ELSE IF (stg_angels.gt.stg_trigger_fl(n,ind1,3)) THEN
        stg_min=stg_min_fc(n,ind1,1)
      ELSE
        IF (runmode.eq.'CALIB') THEN
          stg_min=5.0
        ELSE
          stg_min=5.2
        END IF
      END IF
      idrawdwnc=0
      DO ll=1,idrwn
        IF (n.eq.icanal_no_drawdown(ll)) THEN
          idrawdwnc=1
        END IF
      END DO
      IF (nstorm_day(ndays_sim_sd).eq.1.and.idrawdwnc.eq.1) THEN
        stg_min=stg_min-1.0
      END IF
      idualopsc=0
      DO ll=1,ndual_canal_ops(1)
        IF (n.eq.icanal_no_dualops(ll,1)) THEN
          idualopsc=1
        END IF
      END DO
      IF (nstorm_day(ndays_sim_sd).ne.1.and.idualopsc.eq.1) THEN
        stg_min=stg_min-0.5
      END IF
C
      cout_fc=0.0
      DO ns=1,nsubstr(n,ind1)-1
        IF (ipurpse(n,ind1,ns).ne.1) THEN
          IF (logdisg) THEN
C
            IF (chdep(n).gt.stg_min+0.4) THEN
              coutfl=str_cap(ns)
            ELSE IF (chdep(n).gt.stg_min) THEN
              pctgo=6.25*(chdep(n)-stg_min)**2
              coutfl=pctgo*str_cap(ns)
```

```
                ELSE
                  coutfl=0.0
                END IF
C
              ELSE
                coutfl=0.0
              END IF
            ELSE
              coutfl=0.0
            END IF
            cout_fc=cout_fc+coutfl
          END DO
          ilast_subs=nsubstr(n,ind1)
          IF (ingvhdopt(n,ind1,ilast_subs).eq.1) THEN
            IF (hw.le.tw) THEN
              capacp=discoef(n,ind1,ilast_subs,imgt)*frac_struc_cap(n,
     &         ind1,month)
            ELSE
              capacp=0.0
            END IF
          ELSE
            capacp=discoef(n,ind1,ilast_subs,imgt)*frac_struc_cap(n,
     &       ind1,month)
          END IF
          IF (nint(dqu(idncl)).gt.0) THEN
            avvolq=(chdep(n)-stgintke(n,ind1,ilast_subs,imgt))*cl(n)*
     &       width(n)+cgseep(n,stgintke(n,ind1,ilast_subs,imgt),0,0.)+
     &       tovflc(n,stgintke(n,ind1,ilast_subs,imgt),0,0.,iter,1)+
     &       qu(n)
            avvolq=max(avvolq,0.0)/86400.
            capac=min(capacp,avvolq)
            avvol_fc=(chdep(n)-stg_min)*cl(n)*width(n)+cgseep(n,stg_min,
     &       0,0.)+tovflc(n,stg_min,0,0.,iter,1)+qu(n)
            avvol_fc=max(avvol_fc,0.0)/86400.
            cout_fc=cout_fc+avvol_fc
          ELSE
            IF (ipurpse(n,ind1,ilast_subs).ne.1) THEN
              IF (logdisg) THEN
                avvolq=(chdep(n)-stg_min)*cl(n)*width(n)+cgseep(n,
     &           stg_min,0,0.)+tovflc(n,stg_min,0,0.,iter,1)+qu(n)
                avvolq=max(avvolq,0.0)/86400.
                capac=min(avvolq,capacp)
              ELSE
                capac=0.0
              END IF
            ELSE
              capac=0.0
            END IF
          END IF
          str_cap(ilast_subs)=amin1(capac,capacp)
          IF (conv.eq.1) s331_pump=str_cap(ilast_subs)
          capac_ws=0.0
          DO ns=1,nsubstr(n,ind1)
            IF (ipurpse(n,ind1,ns).eq.1.or.ipurpse(n,ind1,ns).eq.3)
     &       THEN
              capac_ws=capac_ws+str_cap(ns)
            END IF
```

```
          END DO
          IF (nint(dqu(idncl)).gt.0) THEN
            flow_ds=amax1(cout_fc,dqu(idncl))
            IF (tdmnd.gt.0.0) THEN
              ratio=tempvol/tdmnd
            ELSE
              ratio=0.0
            END IF
            wratio(n)=amin1(ratio,1.0)
            IF (nprty(n,ind1).eq.0) THEN
              flow_ds=amax1(cout_fc,dqu(idncl)*wratio(n))
              IF (cout_fc.gt.0.0) THEN
                cout=amin1(flow_ds,capac_ws)
              ELSE
                cout=amin1(capac_ws,wratio(n)*dqu(idncl))
              END IF
              tdmnd=tdmnd-dqu(idncl)
            ELSE
              cout=amax1(flow_ds,tempvol)
              cout=amin1(cout,capac_ws)
            END IF
            IF (str_cap(ilast_subs).gt.0.0.and.conv.eq.1) THEN
              IF (dqu(idncl).lt.str_cap(1)) THEN
                s331_pump=0.0
              ELSE
                s331_pump=cout
              END IF
            END IF
            tempvol=tempvol-amin1(cout,tempvol)
          ELSE
            cout=cout_fc+str_cap(ilast_subs)
          END IF

          IF (conv.eq.1.) THEN
c           print *,chdep(n),cout,stg_min,avvolq,avvol_fc,logdisg
c    $       ,'  S331 ',chdep(idncl),chdep(lc111),chdep(lc111e)
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            s331ws=amin1(cout,dqu(idncl))
            s331fc=cout-amin1(cout,dqu(idncl))
            dqu(idncl)=dqu(idncl)-amin1(cout,dqu(idncl))
            kflo(iflo)=cout
          END IF
C-----------------------------------------------------
C                          S-197
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(68)) THEN
          avvolq=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvolq=max(avvolq,0.0)/86400.
          IF (chdep(n).gt.rmnstgs(n,ind1,iseas)) THEN
            frac_go=(chdep(n)-rmnstgs(n,ind1,iseas))**2/(rmxstg(n,ind1,
     &       iseas)-rmnstgs(n,ind1,iseas))**2
            frac_go=amin1(frac_go,1.0)
          ELSE
            frac_go=0.0
```

```
          END IF
C
          tw197=get_tide(ycnlnode(n,1),jday,proj_sea_level_rise)
          tw = 1.025*(tw197+2.0)-2.0
          IF (earth_plug_opt.eq.'NO ') THEN
            IF (chdep(n).gt.tw) THEN
              cout_max_cap=frac_go*discoef(n,ind1,1,1)*(chdep(n)-tw)**
     &         pwr(n,ind1,1,1)*frac_struc_cap(n,ind1,month)
              DO ic=1,n_trigger_fc_cnl(n,ind1)
                IF (chdep(icnl_no_fc(n,ind1,ic)).gt.stg_trig_oper(n,
     &           ind1,ic,1)) THEN
                  frac_full=7.0/13.0+((min(chdep(icnl_no_fc(n,ind1,ic)
     &             )-stg_trig_oper(n,ind1,ic,1),0.1))**2/(0.1)**2)*(6.0/
     &             13.0)
                  cout=amin1(frac_full,frac_go)*discoef(n,ind1,1,1)*
     &             (chdep(n)-tw)**pwr(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &             month)
                  IF (lflowresid) THEN
                    IF (nint(dqu(lc111)).eq.0) THEN
                      cout=max(cout,kflo(ifls176))
                      cout=min(cout_max_cap,cout)
                    END IF
                  END IF
                  IF (conv.eq.1.) lflowresid=.true.
                  GO TO 80
                END IF
              END DO
              DO ic=1,n_trigger_fc_cnl(n,ind1)
                IF (chdep(icnl_no_fc(n,ind1,ic)).gt.stg_trig_oper(n,
     &           ind1,ic,2)) THEN
                  frac_full=3.0/13.0+((chdep(icnl_no_fc(n,ind1,ic))-
     &             stg_trig_oper(n,ind1,ic,2))**2/(0.5*(stg_trig_oper(n,
     &             ind1,ic,1)-stg_trig_oper(n,ind1,ic,2))**2))*(4.0/
     &             13.0)
                  frac_full=amin1(frac_full,7.0/13.0)
                  cout=amin1(frac_full,frac_go)*discoef(n,ind1,1,1)*
     &             (chdep(n)-tw)**pwr(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &             month)
                  IF (lflowresid) THEN
                    IF (nint(dqu(lc111)).eq.0) THEN
                      cout=amax1(cout,kflo(ifls176))
                      cout=amin1(cout_max_cap,cout)
                    END IF
                  END IF
                  IF (conv.eq.1.) lflowresid=.true.
                  GO TO 80
                END IF
              END DO
              DO ic=1,n_trigger_fc_cnl(n,ind1)
                IF (chdep(icnl_no_fc(n,ind1,ic)).gt.stg_trig_oper(n,
     &           ind1,ic,3)) THEN
                  frac_full=((chdep(icnl_no_fc(n,ind1,ic))-
     &             stg_trig_oper(n,ind1,ic,3))**2/(0.5*(stg_trig_oper(n,
     &             ind1,ic,2)-stg_trig_oper(n,ind1,ic,3))**2))*(3.0/
     &             13.0)
                  frac_full=amin1(frac_full,3.0/13.0)
                  cout=amin1(frac_full,frac_go)*discoef(n,ind1,1,1)*
```

```
     &             (chdep(n)-tw)**pwr(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &             month)
                  IF (lflowresid) THEN
                    IF (nint(dqu(lc111)).eq.0) THEN
                      cout=amax1(cout,kflo(ifls176))
                      cout=amin1(cout_max_cap,cout)
                    END IF
                  END IF
                  IF (conv.eq.1.) lflowresid=.true.
                  GO TO 80
                END IF
              END DO
              IF (nint(dqu(lc111)).eq.0) THEN
                IF (kflo(ifls176).lt.500.0) lflowresid=.false.
                IF (lflowresid) THEN
                  cout=amin1(cout_max_cap,kflo(ifls176))
                ELSE
                  cout=0.0
                END IF
              ELSE
                cout=0.0
                lflowresid=.false.
              END IF
C          if(conv.eq.1) print *,cout,chdep(n),chdep(lc111e),'  DDD'
            ELSE
              cout=0.0
              frac_full=0.0
C          if(conv.eq.1) print *,chdep(n),chdep(lc111e),tw,cout,'  EEE'
            END IF
          ELSE
            IF (chdep(n).gt.tw) THEN
              IF (chdep(icnl_no_fc(n,ind1,1)).gt.stg_trig_oper(n,ind1,1,
     &         1).and.chdep(icnl_no_fc(n,ind1,2)).ge.stg_trig_oper(n,
     &         ind1,2,1)) THEN
                cout=frac_go*discoef(n,ind1,1,1)*(chdep(n)-tw)**pwr(n,
     &           ind1,1,1)*frac_struc_cap(n,ind1,month)
              ELSE
                frac_full=3.0/13.0
                cout=amin1(frac_full,frac_go)*discoef(n,ind1,1,1)*
     &           (chdep(n)-tw)**pwr(n,ind1,1,1)*frac_struc_cap(n,ind1,
     &           month)
              END IF
            ELSE
              cout=0.0
            END IF
          END IF
C999     if (conv.eq.1)
C    $   print *,chdep(n),tw,cnm(lc111e),cnm(lc111)
C    $    ,cout,avvolq,LFLOWRESID,flnm(ifls176),'  S197 ',FLNM(IFLO)
C    $    ,rmnstgs(n,ind1,iseas)
   80     cout=min(cout,avvolq)
C-----------------------------------------------------
C   WATER SUPPLY FROM C-51W TO ACME
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(91)) THEN
          IF (chdep(ica1) .le. floor_elev(iwca1,1)) THEN
             cout = min(dqu(idncl),discoef(n,ind1,1,1))
```

```
          ELSE
             cout = 0.0
          ENDIF
          if (conv.eq.1.) dqu(idncl) = dqu(idncl) - cout
C-----------------------------------------------------
C   WATER SUPPLY FROM C-51 TO ACME
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(92)) THEN
          IF (chdep(ica1) .le. floor_elev(iwca1,1)) THEN
             cout = min(dqu(idncl),discoef(n,ind1,1,1))
          ELSE
             cout = 0.0
          ENDIF
          if (conv.eq.1.) dqu(idncl) = dqu(idncl) - cout
C-----------------------------------------------------
C   PROPOSED PUMP : C-18 TO LOX SLOUGH
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(80)) THEN
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
          avvol_up=max(avvol_up/86400.,0.0)
          IF (tdmnd.gt.0.0) THEN
            ratio_ws=amin1(tempvol/tdmnd,1.0)
          ELSE
            ratio_ws=0.0
          END IF
          ratio_ws_loxsl=min(avvol_up/discoef(n,ind1,1,1),1.0)
          IF (loxenv.eq.'FALSE') THEN
            cout=0.0
          ELSE
            IF (lwslo(iloxah)) THEN
              cout=discoef(n,ind1,1,1)*amin1(ratio_ws,ratio_ws_loxsl)
            ELSE
              cout=0.0
            END IF
          END IF
          tempvol=tempvol-cout
          IF (lwslo(iloxah)) THEN
            tdmnd=tdmnd-discoef(n,ind1,1,1)
          END IF
          IF (conv.eq.1) THEN
            pond(loxnod1)=pond(loxnod1)+cout*86400./(2.*gdar
     &      *(1.0-carfact(loxnod1)-total_small_afact(loxnod1)))
            pond(loxnod2)=pond(loxnod2)+cout*86400./(2.*gdar
     &      *(1.0-carfact(loxnod2)-total_small_afact(loxnod2)))
            kflo(iflo)=cout
          END IF
C
C-----------------------------------------------------
C                          S-316
C                       PROPOSED STRCTR
C-----------------------------------------------------
        ELSE IF (flnm(iflo).eq.strname_spec(69)) THEN
          avvol_up=(chdep(n)-rmnstgs(n,ind1,iseas))*cl(n)*width(n)+
     &     cgseep(n,rmnstgs(n,ind1,iseas),0,0.)+tovflc(n,rmnstgs(n,ind1,
     &     iseas),0,0.,iter,1)+qu(n)
```

```
          avvol_up=max(avvol_up/86400.,0.0)
          IF (dqu(idncl).gt.0.0) THEN
            cout=min(avvol_up,discoef(n,ind1,1,1),dqu(idncl))
            wsneedl8s=dqu(il8s)-cout
            avvol_up=avvol_up-cout
            IF (conv.eq.1.) THEN
              IF (idncl.gt.n) THEN
                IF (iserv_area_loc(idncl).gt.0) THEN
                  rloc_excess_for_ws(iserv_area_loc(idncl))=
     &             rloc_excess_for_ws(iserv_area_loc(idncl))+amin1(cout,
     &             dqu(idncl))*86400.
                END IF
              ELSE
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+cout*86400.
              END IF
            END IF
          ELSE
            cout=min(avvol_up,discoef(n,ind1,1,1))
            IF (idncl.lt.n.and.conv.eq.1.) THEN
              rloc_excss_to_canl_next_step(idncl)=
     &         rloc_excss_to_canl_next_step(idncl)+cout*86400.
            END IF
            wsneedl8s=0.0
          END IF
C
C  if runoff from L-8 north of S-316 is insufficient to meet needs
C  downstream,then supply water from WPB canal
C
          avcaps352=s352mx-kflo(iflhg5)
          IF (agp(is5absn-6).ge.0.0) THEN
C
C  available capacity in canal
C
            avcnlcap=fllkwpb-kflo(iflhg5)
            avcnlcap=amax1(avcnlcap,0.0)
            avcap=amin1(avcaps352,avcnlcap)
C
C
            IF (stagelo.lt.sched(iday,month)) THEN
              IF (add_dmndagl8.gt.0.00001) THEN
                wsdel=amin1(wsneedl8s,add_dmndagl8/1.9835)*pctmet
                wsdelfrwpb=amin1(wsdel,avcap,700.0)
              ELSE
                wsdelfrwpb=0.0
              END IF
            ELSE
              wsdelfrwpb=amin1(avcap,wsneedl8s,700.0)
            END IF
C             if (conv .eq. 1)
C    $        print *,wsdelfrwpb,fllkwpb,wsneedl8s,kflo(iflhg5)
C    $          ,s352mx , avcap,avcaps352,avcnlcap,month,iday
C    $          ,agp(is5absn-6)/86400.,wss352
            IF (conv.eq.1) THEN
              IF (lok_split_option.eq.0) THEN
                finstor=finstor-wsdelfrwpb*1.9835
                stagelo=stg2sto(finstor,1)
```

```
              ELSE
                IF (stagelo_beg.gt.stgref.or.stagelo_res_beg.lt.2.0)
     &           THEN
                  finstor=finstor-wsdelfrwpb*1.9835
                  stagelo=stg2stolitz(finstor,1)
                  wsdelfrwpb_res=0.0
                ELSE
                  finstor_res=finstor_res-wsdelfrwpb*1.9835
                  stagelo_res=stg2stores(finstor_res,1)
                  wsdelfrwpb_res=wsdelfrwpb
                END IF
              END IF
              wsneedl8s=wsneedl8s-wsdelfrwpb
              kflo(iflhg5)=kflo(iflhg5)+wsdelfrwpb
              kflo(ifll8)=kflo(ifll8)-wsdelfrwpb
              kflo(ifls5a)=kflo(ifls5a)+wsdelfrwpb
              ws_to_l8_via_s352=wsdelfrwpb
              s5a3_no=s5a3_no+wsdelfrwpb
              qu(idncl)=qu(idncl)+wsdelfrwpb*86400.
            END IF
          ELSE
            wsdelfrwpb=0.0
          END IF
          wsneed_iter=wsneedl8s-wsdelfrwpb
C            if (conv .eq. 1) then
C             write(445,*) wsdelfrwpb,wsneedl8s,avcap,agp(is5absn-6)
C    $             /86400.,'   first'
C            endif
C
C  if wpb canal cannot satisfy needs,then take water from
C  L-8 north of S-316
C
          IF (wsneed_iter.gt.0.000001) THEN
            stgdnc=chdep(idncl)+0.1
            avvol_ws=(chdep(n)-stgdnc)*cl(n)*width(n)+cgseep(n,stgdnc,0,
     &       0.)+tovflc(n,stgdnc,0,0.,iter,1)+qu(n)
            avvol_ws=amax1(avvol_ws,0.0)/86400.
            coutws=amin1(discoef(n,ind1,1,1)-cout,wsneed_iter,avvol_ws)
            cout=cout+coutws
          ELSE
            coutws=0.0
          END IF
          IF (conv.eq.1) wsneedl8s=wsneedl8s-coutws
          wsneed_iter=dqu(il8s)-cout-wsdelfrwpb
c         print *,wsneed_iter,dqu(il8s),cout,coutws,wsdelfrwpb
c    &    ,'  WSNEEDITER'
C
          IF (stagelo.gt.chdep(il8w)+hdc(il8w,iseas)) THEN
            capac=645.*sqrt(stagelo-chdep(il8w)-hdc(il8w,iseas))
          ELSE
            capac=0.0
          END IF
          IF (nint(wsneed_iter).gt.0.or.nint(dqu(il8w)).gt.0) THEN
            lpmps309=.false.
            total_needs=wsneed_iter+dqu(il8w)
            IF (dmndagl8+add_dmndagl8.gt.0.000001) THEN
              IF (nint(wsneed_iter).gt.0.and.nint(dqu(il8w)).gt.0) THEN
```

```
              total_ag_dmnd=dmndagl8/1.9835 + add_dmndagl8/1.9835
            ELSE
              total_ag_dmnd=dmndagl8/1.9835
            END IF
          ELSE
            total_ag_dmnd=0.0
          END IF
          IF (stagelo.lt.sched(iday,month)) THEN
            IF (dmndagl8+add_dmndagl8.gt.0.000001) THEN
              wsdel=amin1(total_needs,total_ag_dmnd)*pctmet
              coutc10a=amin1(wsdel,capac)
            ELSE
              coutc10a=0.0
            END IF
          ELSE
            wsdel=total_needs
            coutc10a=amin1(wsdel,capac)
          END IF
          IF (conv.eq.1.) THEN
            IF (nint(dqu(il8w)).eq.0) THEN
              wsneedl8s=wsneedl8s-coutc10a
              iagflo(iag(il8agbsn),1)=-nint(amin1(total_ag_dmnd,
     &         wsneedl8s))
            ELSE IF (nint(wsneed_iter).eq.0) THEN
              iagflo(iag(il8agbsn),1)=-nint(amax1(amin1(dqu(il8w),
     &         total_ag_dmnd)-coutc10a,0.0))
            ELSE
              frac_met=coutc10a/total_needs
              iagflo(iag(il8agbsn),1)=-nint((1.0-frac_met)*
     &         total_ag_dmnd)
            END IF
          END IF
        ELSE
          coutc10a=0.0
          lpmps309=.true.
        END IF
C
        IF (dmndagl8.gt.0.0.and.opt_l8_prop.and.conv.eq.1) THEN
          IF (nint(dqu(il8w)).gt.0.and.nint(wsneed_iter).gt.0) THEN
            qu(il8w)=qu(il8w)-(dmndagl8/1.9835-
     &       abs(iagflo(iag(il8agbsn),1))*0.5)*86400.
            qu(il8s)=qu(il8s)-(add_dmndagl8/1.9835-
     &       abs(iagflo(iag(il8agbsn),1))*0.5)*86400.
          ELSE IF (nint(dqu(il8w)).gt.0.and.nint(wsneed_iter).eq.0)
     &     THEN
            qu(il8w)=qu(il8w)-(dmndagl8/1.9835-
     &       abs(iagflo(iag(il8agbsn),1)))*86400.
            qu(il8s)=qu(il8s)-add_dmndagl8*43560.
          ELSE IF (nint(dqu(il8w)).eq.0.and.nint(wsneed_iter).gt.0)
     &     THEN
            qu(il8w)=qu(il8w)-dmndagl8*43560.
            qu(il8s)=qu(il8s)-(add_dmndagl8/1.9835-
     &       abs(iagflo(iag(il8agbsn),1)))*86400.
          ELSE
            qu(il8w)=qu(il8w)-dmndagl8*43560.
            qu(il8s)=qu(il8s)-add_dmndagl8*43560.
          END IF
```

```
c           print *,qu(il8s)/86400.,add_dmndagl8/1.9835,
c    &       iagflo(iag(il8agbsn),1),'  MNMNB'
          END IF
C
          IF (conv.eq.1) THEN
            qu(il8w)=qu(il8w)+coutc10a*86400.
            dqu(il8w)=dqu(il8w)-amin1(coutc10a,dqu(il8w))
            avvol_up=(chdep(il8w)-chdep(n))*cl(il8w)*width(il8w)+
     &       tovflc(il8w,chdep(n),0,0.,iter,2)+cgseep(il8w,chdep(n),0
     &       ,0.)+qu(il8w)
            avvol_up=max(avvol_up,0.0)/86400.
            qu(il8w)=qu(il8w)-avvol_up*86400.
            upstream_inflow(il8w)=upstream_inflow(il8w)+(coutc10a-
     &       avvol_up)*86400.
            qu(idncl)=qu(idncl)+avvol_up*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+avvol_up*
     &       86400.
            kflo(iflc10a)=coutc10a+kflo(iflc10a)
            IF (lok_split_option.eq.0) THEN
              finstor=finstor-coutc10a*1.9835
              stagelo=stg2sto(finstor,1)
            ELSE
              IF (stagelo_beg.gt.stgref.or.stagelo_res_beg.lt.2.0) THEN
                finstor=finstor-coutc10a*1.9835
                stagelo=stg2stolitz(finstor,1)
                l8ws_res=0.0+l8ws_res
              ELSE
                finstor_res=finstor_res-coutc10a*1.9835
                stagelo_res=stg2stores(finstor_res,1)
                l8ws_res=coutc10a+l8ws_res
              END IF
            END IF
          END IF
C           if (conv .eq. 1)
C    $       print *,coutc10a,avvol_up,chdep(il8w),chdep(n),qu(il8w)
C    $                 /86400.,'   in wroute'
C
c            print *,COUT,COUTC10A,CHDEP(N),wsdelfrwpb,rf316tol8
c    $         ,CHDEP(IDNCL),coutws,month,iday,ifyr+iyear-1,'  S316'
          IF (conv.eq.1) THEN
            qu(idncl)=qu(idncl)+cout*86400.
            upstream_inflow(idncl)=upstream_inflow(idncl)+cout*86400.
            dqu(idncl)=dqu(idncl)-amin1(cout,dqu(idncl))
            kflo(iflo)=cout
C            write(445,*),COUT,COUTC10A,CHDEP(N),wsdelfrwpb,rf316tol8
C    $         ,CHDEP(IDNCL),coutws,month,iday,ifyr+iyear-1
          END IF
C
C-----------------------------------------------------------------
C              RUNOFF FROM L-8 TO WCA1 AND C-51W
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.strname_spec(58)) THEN
          tot_pump=0.0
          rmin_fc_stg=rmnstgs(n,ind1,iseas)
          pmptomcnl=strout(rmin_fc_stg,n,n,1,1,crmin,volwtr,prevqu,rain,
     &     iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
```

```
      IF (.not.opt_l8_prop) THEN
        tot_pump=0.0
        bkpmptolofrl8=strout(rmin_fc_stg,n,n,2,1,crmin,volwtr,
     &   prevqu,rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,
     &   0)
      ELSE
        bkpmptolofrl8=0.0
      END IF
C
      avvol_up=(chdep(n)-rmin_fc_stg)*cl(n)*width(n)+cgseep(n,
     &  rmin_fc_stg,0,0.)+tovflc(n,rmin_fc_stg,0,0.,iter,1)-
     &  bkpmptolofrl8*86400+qu(n)-pmptomcnl*86400.
      avvol_up_1=max(avvol_up/86400.,0.0)
      avvol_up=max(avvol_up_1-flow_to_l8sta,0.0)
      IF (nres(n,ind1).gt.0) THEN
        rem_capac_l8bpit=amax1(amin1(weircff(irsrcvr(n,ind1,1),1,1),
     &   volwtr(irsrcvr(n,ind1,1)))-tot_avl_l8_bpit_res,0.0)
        avvol_up_2=avvol_up-min(avvol_up,rem_capac_l8bpit)
        rl8runoff_to_bptres=min(avvol_up,rem_capac_l8bpit)
        IF (conv.eq.1.) THEN
          IF (sfactor(irsrcvr(n,ind1,1))
     &     .ge.sfactmin.or.ires_small_sim(irsrcvr(n,ind1,1)).eq.'NO
     &') THEN
            pond(iupstrm(irsrcvr(n,ind1,1),1,1))=
     &       pond(iupstrm(irsrcvr(n,ind1,1),1,1))+min(avvol_up,
     &       rem_capac_l8bpit)*86400./(gdar*(1.0-carfact
     &       (iupstrm(irsrcvr(n,ind1,1),1,1))-total_small_afact
     &       (iupstrm(irsrcvr(n,ind1,1),1,1))))
            volwtr(irsrcvr(n,ind1,1))=volwtr(irsrcvr(n,ind1,1))-
     &       min(avvol_up,rem_capac_l8bpit)
          ELSE
            if (NINSTR(irsrcvr(n,ind1,1)) .gt. 0) THEN
              nodrcv_s=iupstrm(irsrcvr(n,ind1,1),1,1)
            else
              if (IRCEIVE(n,ind1) .eq. 2) then
                nodrcv_s=nodrcv(n,ind1,1)
              else
                nodrcv_s=max_ncells
              endif
            endif
            DO kk=1,noresincell(nodrcv_s)
              IF (irsrcvr(n,ind1,1).eq.ires_index_in_cell(nodrcv_s,
     &         kk)) THEN
                IF (stage_res(nodrcv_s,kk).gt.ells_in_res(nodrcv_s,
     &           kk)) THEN
                  pond_in_res(nodrcv_s,kk)=pond_in_res(nodrcv_s,kk)+
     &             rl8runoff_to_bptres*86400./(sfactor(irsrcvr(n,
     &             ind1,1))*gdar)
                  stage_res(nodrcv_s,kk)=ells_in_res(nodrcv_s,kk)+
     &             pond_in_res(nodrcv_s,kk)
                ELSE
                  dpth_below_ls_res=(ells_in_res(nodrcv_s,kk)-
     &             stage_res(nodrcv_s,kk))*s(nodrcv_s)
                  dpth_of_inflw=rl8runoff_to_bptres*86400./
     &             (sfactor(irsrcvr(n,ind1,1))*gdar)
                  stage_res(nodrcv_s,kk)=stage_res(nodrcv_s,kk)+
     &             amin1(dpth_of_inflw,dpth_below_ls_res)/
```

```
     &                 s(nodrcv_s)+amax1(dpth_of_inflw-
     &                 dpth_below_ls_res,0.0)
                      pond_in_res(nodrcv_s,kk)=amax1(stage_res(nodrcv_s,
     &                 kk)-ells_in_res(nodrcv_s,kk),0.0)
                    END IF
                    res_struc_inflw(irsrcvr(n,ind1,1))=
     &               res_struc_inflw(irsrcvr(n,ind1,1))+
     &               rl8runoff_to_bptres*1.9835
                  END IF
                END DO
              END IF
            END IF
            avvol_up=avvol_up_2
          END IF
C
C    RUNOFF TO WCA-1
C
          tot_pump=0.0
          pmptomcnl=strout(chdep(ica1),n,n,1,1,crmin,volwtr,prevqu,rain,
     &     iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)
          IF (.not.opt_l8_prop) THEN
            tot_pump=0.0
            bkpmptolofrl8=strout(chdep(ica1),n,n,2,1,crmin,volwtr,
     &       prevqu,rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,
     &       0)
          ELSE
            bkpmptolofrl8=0.0
          END IF
C
          IF (.not.opt_l8_prop.and.istaopt(is5absn-6).eq.0) THEN
            avvoltoca1=(chdep(n)-chdep(ica1))*cl(n)*width(n)+cgseep(n,
     &       chdep(ica1),0,0.)+tovflc(n,chdep(ica1),0,0.,iter,1)-
     &       bkpmptolofrl8*86400+qu(n)-pmptomcnl*86400.
            avvoltoca1=max(avvoltoca1/86400.,0.0)
            IF (chdep(n).gt.chdep(ica1)+0.25) THEN
              caps5a2=capacs5as
            ELSE IF (chdep(n).gt.chdep(ica1)) THEN
              caps5a2=capacs5as/0.25*(chdep(n)-chdep(ica1))
            ELSE
              caps5a2=0.0
            END IF
            coutca1=min(avvol_up,caps5a2,avvoltoca1)
            cout=coutca1+rl8runoff_to_bptres
            flfrml8toca1=coutca1
            IF (conv.eq.1) THEN
              qu(ica1)=qu(ica1)+coutca1*86400.
              kflo(ifll8)=kflo(ifll8)+coutca1
              kflo(ifls5a5)=kflo(ifls5a5)+coutca1
            END IF
          END IF
C
C    RUNOFF TO WESTERN C-51
C
          avalc51w=avvol_up-coutca1
C
          IF (.not.opt_l8_prop.and.istaopt(is5absn-6).eq.0) THEN
            volout=0.0
```

```
            tot_pump=0.0
            DO ns=1,nstrctr(idncl)
              dble13 = dble(13.0)
              volout=strout(dble13,idncl,idncl,ns,0,crmin,volwtr,prevqu,
     &         rain,iter,tot_pump,jday,tempf_volwtr,tempf_weircff,0)+
     &         volout
            END DO
            volout=volout*86400.
            avvolin=(dble13-hdc(idncl,iseas)-chdep(idncl))*cl(idncl)*
     &       width(idncl)-cgseep(idncl,13.0-hdc(idncl,iseas),0,0.)-
     &       tovflc(idncl,dble13-hdc(idncl,iseas),0,0.,iter,2)-drainc51w
     &       +volout
            avvolin=max(avvolin,0.0)/86400.
            coutc51w=min(avalc51w,avvolin)
            IF (chdep(n).gt.chdep(idncl)+hdc(idncl,iseas)) THEN
              capac_e_s5a4=553.*(chdep(n)-chdep(idncl)-hdc(idncl,iseas))*
     &         *0.5
            ELSE
              capac_e_s5a4=0.0
            END IF
c           IF (chdep(n).gt.chdep(idncl)+hdc(idncl,iseas)+0.25) THEN
c             capac_e_s5a4=s5a4max
c           ELSE IF (chdep(n).gt.chdep(idncl)+hdc(idncl,iseas)) THEN
c             capac_e_s5a4=s5a4max*4.0*(chdep(n)-chdep(idncl)-hdc(idncl,
c    &         iseas))
c           ELSE
c             capac_e_s5a4=0.0
c           END IF
            s5a4av=amax1(capac_e_s5a4-kflo(iflbel5),0.0)
            IF (chdep(n).gt.chdep(idncl)+hdc(idncl,iseas)) THEN
              IF (coutc51w.le.s5a4av) THEN
                cout=coutca1+coutc51w+rl8runoff_to_bptres
                flfrml8toc51w=coutc51w
                IF (conv.eq.1) THEN
                  kflo(iflbel5)=coutc51w+kflo(iflbel5)
                  IF (idncl.gt.n) THEN
                    IF (iserv_area_loc(idncl).gt.0) THEN
                      rloc_excess_for_ws(iserv_area_loc(idncl))=
     &                 rloc_excess_for_ws(iserv_area_loc(idncl))+
     &                 amin1(coutc51w,dqu(idncl))*86400.
                    END IF
                  ELSE
                    rloc_excss_to_canl_next_step(idncl)=
     &               rloc_excss_to_canl_next_step(idncl)+coutc51w*
     &               86400.
                  END IF
                  qu(idncl)=qu(idncl)+coutc51w*86400.
                  upstream_inflow(idncl)=upstream_inflow(idncl)+
     &             coutc51w*86400.
                  dqu(idncl)=dqu(idncl)-coutc51w
                  dqu(idncl)=amax1(dqu(idncl),0.0)
                END IF
              ELSE
                cout=coutca1+s5a4av+rl8runoff_to_bptres
                flfrml8toc51w=s5a4av
                IF (conv.eq.1) THEN
                  kflo(iflbel5)=s5a4av+kflo(iflbel5)
```

```
                  qu(idncl)=qu(idncl)+s5a4av*86400.
                  IF (idncl.gt.n) THEN
                    IF (iserv_area_loc(idncl).gt.0) THEN
                      rloc_excess_for_ws(iserv_area_loc(idncl))=
     &                 rloc_excess_for_ws(iserv_area_loc(idncl))+
     &                 amin1(s5a4av,dqu(idncl))*86400.
                    END IF
                  ELSE
                    rloc_excss_to_canl_next_step(idncl)=
     &               rloc_excss_to_canl_next_step(idncl)+s5a4av*86400.
                  END IF
                  upstream_inflow(idncl)=upstream_inflow(idncl)+s5a4av*
     &             86400.
                  dqu(idncl)=dqu(idncl)-s5a4av
                  dqu(idncl)=amax1(dqu(idncl),0.0)
                END IF
              END IF
              IF (conv.eq.1) THEN
                kflo(ifll8)=kflo(ifll8)+flfrml8toc51w
                s5a3_so = s5a3_so + flfrml8toc51w
              END IF
            ELSE
              flfrml8toc51w=0.0
            END IF
C
C  ANY REMAINING RUNOFF TO BE PUMPED THRU S-5A AFTER S5A BASIN RUNOFF IS
C  REMOVED
            aval_rem_runoff=avvol_up-cout
            rem_capac_s5a=s5amax-kflo(ifls5a)
            couts5a=amin1(aval_rem_runoff,rem_capac_s5a,700.)
            cout=coutca1+flfrml8toc51w+couts5a+rl8runoff_to_bptres
            IF (conv.eq.1) THEN
              flfrml8toca1=flfrml8toca1+couts5a
              qu(ica1)=qu(ica1)+couts5a*86400.
              kflo(ifll8)=kflo(ifll8)+couts5a
              kflo(ifls5a5)=kflo(ifls5a5)+couts5a
            END IF
C
          ELSE
            volout=0.0
            tot_pump=0.0
            DO ns=1,nstrctr(idncl)
              volout=strout(chdep(n),idncl,idncl,ns,0,crmin,volwtr,
     &         prevqu,rain,iter,tot_pump,jday,tempf_volwtr,
     &         tempf_weircff,0)+volout
            END DO
            avvol_dn=(chdep(n)-chdep(idncl))*cl(idncl)*width(idncl)-
     &       cgseep(idncl,chdep(n),0,0.)-tovflc(idncl,chdep(n),0,0.,iter
     &       ,2)+volout*86400.
            avvol_dn=max(avvol_dn,0.0)/86400.
            IF (chdep(n).gt.chdep(idncl)+hdc(idncl,iseas)) THEN
              capac_s5a4e=553.*(chdep(n)-chdep(idncl)-hdc(idncl,iseas))*
     &         *0.5
            ELSE
              capac_s5a4e=0.0
            END IF
            coutt=min(avvol_up,avvol_dn,capac_s5a4e)
```

```
            cout=coutt+rl8runoff_to_bptres
            IF (conv.eq.1) THEN
              flfrml8toc51w=coutt
              kflo(iflbel5)=coutt+kflo(iflbel5)
              flfrml8toca1=0.0
              s5a3_so = s5a3_so + flfrml8toc51w
              IF (idncl.gt.n) THEN
                IF (iserv_area_loc(idncl).gt.0) THEN
                  rloc_excess_for_ws(iserv_area_loc(idncl))=
     &             rloc_excess_for_ws(iserv_area_loc(idncl))+
     &             amin1(coutt,dqu(idncl))*86400.
                END IF
              ELSE
                rloc_excss_to_canl_next_step(idncl)=
     &           rloc_excss_to_canl_next_step(idncl)+coutt*86400.
              END IF
              dqu(idncl)=dqu(idncl)-coutt
              upstream_inflow(idncl)=upstream_inflow(idncl)+coutt*
     &         86400.
              dqu(idncl)=amax1(dqu(idncl),0.0)
              qu(idncl)=qu(idncl)+coutt*86400.
            END IF
          END IF
        END IF
      END IF
   90 FORMAT (i5,2i3,5x,a6,5x,2f7.2,8f10.0)
C
      RETURN
      END
```

specdssout.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: specdssout.inc,v 1.6 2003/07/03 20:08:05 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/specdssout.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /specdssout/ s333_reg_lec_abve_sch,s333_reg_lec_bel_sch,
     &s356_to_cell,s356_to_l29,rlvseep_to_cnl(max_ncnls),
     &n_str_if_cond(15),n_sta_inf_dss,naddstr_asr_eaa
```

specsetupfdss.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: specsetupfdss.F,v 1.21 2005/06/10 13:29:20 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/specsetupfdss.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE SETS UP SPECIAL FLOW COMPONENTS THAT ARE NOT AUTOMATICALLY
C   CAPTURED IN MODEL TO BE WRITTEN TO DSS FILE (daily_str_flw.dss)
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE specsetupfdss (lok2stluc_est, lok2stluc_irrig,
     & lok2caloos_est, lok2caloos_irrig, holyinq, dmds4, dmds236,
     & s235inf, dmdsemind, lvseep_to_l33, lvseep_to_c6,
     & total_eaa_supply_deliv, water_supply_to_holyl, rlokreg2res1,
     & rlokreg2res2, reg2est1, reg2est2, vol_rec_from_cpbasr, tcnsrnff,
     & tcrnbsres_to_lok, total_reuse, ws_pipeline_3a,
     & ws_pipeline_3a_s150, ws_pipeline_2a, ws_pipeline_ca1,
     & ws_pipeline_ca1_hill, ws_pipeline_ca1_wpb, s31dsg_to_bay, s140fc,
     & fplresws, total_reuse_res, rloc_seep_for_ws, ws_to_l8_via_s352,
     & rloc_excess_for_ws, total_reuse_ws, s333_reg_lec, asr_rec_ws,
     & ws_from_lo_s6, res_outflow_to_basin, ws_from_lo_s5a, rfls155a,
     & outflow_to_canal_roten, outflow_to_marsh, s236so, ws_to_sta5,
     & ws_to_sta6, ws_to_sta34_frm_lok, g155_pos,
     & dischg_to_s8_frm_sta34_adj, dischg_to_s7_frm_sta34_adj,
     & total_flow_g404, wsdel_to_l8, total_add_supply_to_lwdd,
     & wsdel_to_l8s, wsdel_to_c51w, s12_reg_total, s12_env_total,
     & ws_frm_lok_to_semcyp, ws_frm_lok_to_sugh,
     & outflow_from_roten_thru_s8_pres, sem_dmnd_met_bef_lok,
     & flow_from_asr_for_ws_eaa, cfsoth_litz1,
     & flow_from_asr_meet_rem_eaa_dmnd, tot_lok_to_eaares, cfsoth_res1,
     & flow_asr_inj, flow_from_asr_ws, iconvey_canal_no, ires_no,
     & idshg_opt, noutlets, istrctr_ws, divers, ws_to_roten_frm_lok,
```

```
     & nsvarea, asr_rec_ws_cnty, g155_flow_from_3a, isem_flg, nres_ws,
     & jday, cfsoth1, sta34_out_to_wca3ane_adj, ws_from_acmres_lwdd,
     & vol_rec_from_acmasr,rlvseep_to_l30,s4bsn_to_lok, uisrnff,
     & rlokreg_to_l8)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'simcstgs.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'wier.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'ccaloos.inc'
      INCLUDE 'cstlucie.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'flw298dso.inc'
      INCLUDE 'lokres.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'minfllvl.inc'
      INCLUDE 'lvseepdiv.inc'
      INCLUDE 'specdssout.inc'

      INTEGER iconvey_canal_no(max_n_wcas),
     & idshg_opt(max_n_res,max_nostr_res), ires_no(max_n_res),
     & istrctr_ws(max_n_res,max_nostr_res), noutlets(max_n_res)

      REAL asr_rec_ws(max_n_serv_area),
     & asr_rec_ws_cnty(max_n_serv_area), dmdsemind(366),
     & flow_asr_inj(max_n_eaa_condt,max_n_reg_rel_basin),
     & flow_from_asr_ws(max_n_res,max_nostr_res), fplresws(366),
     & lok2caloos_est, lok2caloos_irrig, lok2stluc_est, lok2stluc_irrig,
     & lokreg2caloos, lokreg2stluc, lvseep_divers, lvseep_pump_cap,
     & lvseep_to_c6, lvseep_to_l33,
     & res_outflow_to_basin(max_n_res,max_nostr_res),
     & rloc_excess_for_ws(max_n_serv_area),
     & rloc_seep_for_ws(max_n_serv_area), tcnsrnff(366),
     & total_reuse_ws(max_n_serv_area), uisrnff(366)

      CHARACTER char_id_cnl_ws_resasr*5
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  separate components of flow for s5a2,s5a3,s5a4 for dss

      IF (runmode .EQ. 'CALIB') THEN
         IF (kflo(ifll8) .LT. 0.0) THEN
            s5a3_no = abs(kflo(ifll8))
            s5a3_so = 0.0
         ELSE
```

```
            s5a3_no = 0.0
            s5a3_so = kflo(ifll8)
         ENDIF
      ENDIF

      IF (kflo(iflbel5) .LT. 0.0) THEN
         s5a4_w = abs(kflo(iflbel5))
         s5a4_e = 0.0
      ELSE
         s5a4_w = 0.0
         s5a4_e = kflo(iflbel5)
      ENDIF

      IF (runmode .EQ. 'CALIB') THEN
         IF (kflo(ifls5a5) .LT. 0.0) THEN
            s5a2_no = abs(kflo(ifls5a5))
            s5a2_so = 0.0
         ELSE
            s5a2_no = 0.0
            s5a2_so = kflo(ifls5a5)
         ENDIF
      ENDIF
c
      IF (KFLO(IFLS77) .LT. 0.0) THEN
         S77BAK = ABS(KFLO(IFLS77))
         S77_OUT = 0.0
      ELSE
         S77BAK = 0.0
         S77_OUT = KFLO(IFLS77)
      ENDIF
c
      IF (kflo(IFLS308) .LT. 0.0) THEN
         S308BAK = ABS(KFLO(IFLS308))
         S308_OUT = 0.0
      ELSE
         S308BAK = 0.0
         S308_OUT = KFLO(IFLS308)
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c set lokreg2stluc and lokreg2caloos variables
      IF (runmode .EQ. 'SIMUL') THEN
         lokreg2stluc = S308_OUT-(lok2stluc_est+lok2stluc_irrig)
         lokreg2caloos = S77_OUT-(lok2caloos_est+lok2caloos_irrig)
      ENDIF
C
      IF (kflo(iflagq) .LT. 0.0) THEN
         agqbak = abs(kflo(iflagq))
         agqws = 0.0
      ELSE
         agqbak = 0.0
         agqws = kflo(iflagq)
      ENDIF
c
      IF (kflo(iflpsac) .LT. 0.0) THEN
         acmbak = abs(kflo(iflpsac))
         acmws  = 0.0
      ELSE
```

```
         acmbak = 0.0
         acmws  = kflo(iflpsac)
      ENDIF
c
      IF (dmds236 .LT. 0.0) THEN
         s236_ro = abs(dmds236) / 1.9835
         s236_dmd = 0.0
      ELSE
         s236_ro = 0.0
         s236_dmd = dmds236 / 1.9835
      ENDIF
c
      IF (dmdsugh .LT. 0.0) THEN
         sughro  = abs(dmdsugh)
         sughdmd = 0.0
      ELSE
         sughdmd = dmdsugh
         sughro  = 0.0
      ENDIF
      DO i = 1,no_of_wiers
         ndirect = 0
         DO k = 1,5
            IF (wier_orient(i,k) .NE. 'O') THEN
               ndirect = ndirect + 1
               JSTR1 = JSTR1 + 1
               STRFLW(JSTR1,IDSS) = QWIER(i,ndirect)
            ENDIF
         END DO
      END DO
      n_flow_dss_calib = 0
      IF (runmode .EQ. 'CALIB') THEN
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = QS39TBR
         n_flow_dss_calib = 1
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = HOLYINQ / 86400.
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = s236_ro
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = s236_dmd
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = DMDS4 / 1.9835
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = s4bsn_to_lok / 1.9835
      IF (runmode .EQ. 'SIMUL') THEN
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = SUGHRO / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = SUGHDMD / 1.9835
```

```
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = REGS77
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = REGS308
ENDIF
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S77BAK
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S308BAK
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S77_OUT
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S308_OUT
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = AGQBAK
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = AGQWS
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = ACMBAK
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = ACMWS
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A3_NO
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A3_SO
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A2_NO
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A2_SO
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A4_E
JSTR1 = JSTR1 + 1
n_flow_dss_calib = n_flow_dss_calib + 1
STRFLW(JSTR1,IDSS) = S5A4_W
IF (runmode .EQ. 'SIMUL') THEN
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = REGS354
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = REGS351
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = REGS352
   JSTR1 = JSTR1 + 1
   STRFLW(JSTR1,IDSS) = WSS39
   JSTR1 = JSTR1 + 1
```

```
         STRFLW(JSTR1,IDSS) = WSS34
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = REGS39
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = REGS34
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = FLFRML8TOCA1
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = FLFRML8TOC51W
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s235inf / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = eqs79 / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = eqs80 / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = dmdsemind(jday) / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = nnrreg
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = hlsbreg
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s8_pump_flow
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s8_grav_flow
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s7_pump_flow
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s7_grav_flow
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s331_pump
      ENDIF
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = lvseep_to_l33
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = lvseep_to_c6
c
      IF (KFLO(IFLS2) .LT. 0) THEN
         S2PUMP =  - KFLO(IFLS2)
         S351SPW = 0.0
      ELSE
         S2PUMP = 0.0
         S351SPW = KFLO(IFLS2)
      ENDIF
C
      IF (KFLO(IFLS3) .LT. 0) THEN
         S3PUMP =  - KFLO(IFLS3)
         S354SPW = 0.0
      ELSE
         S3PUMP = 0.0
         S354SPW = KFLO(IFLS3)
      ENDIF
c
c
      JSTR1 = JSTR1 + 1
```

```
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = S2PUMP
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = S351SPW
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = S3PUMP
      JSTR1 = JSTR1 + 1
      n_flow_dss_calib = n_flow_dss_calib + 1
      STRFLW(JSTR1,IDSS) = S354SPW
      IF (runmode .eq. 'CALIB') THEN
         JSTR1 = JSTR1 + 1
         n_flow_dss_calib = n_flow_dss_calib + 1
         STRFLW(JSTR1,IDSS) = rlvseep_to_l30
      ENDIF
      IF (runmode .EQ. 'CALIB') THEN
         IF (n_str_if_cond(1) .ne. n_flow_dss_calib) THEN
              WRITE(*,1500) n_str_if_cond(1),n_flow_dss_calib
 1500         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH) UNCONDITIONALLY NOT MATCHING NO OF VALUES IN CODE. NO OF NAMES
     $OUTPUT = ',I3,'NO OF DATA VALUES IN CODE = ',I3,'. WILL HAVE MISMA
     $TCH.CHECK INPUT  NEAR BOTTOM OF canal_struc_specs.dat FILE.')
              STOP
         ENDIF
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (runmode .EQ. 'SIMUL') THEN
         JSTR1 = JSTR1 + 1
         S352SO = KFLO(IFLHG5)
         STRFLW(JSTR1,IDSS) = S352SO
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = WSS354
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = WSS351
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = WSS352
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_to_lec_s354
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_to_lec_s351
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_to_lec_s352
c  make additional WRITEs to str2x2.dss
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = total_eaa_supply_deliv / 1.9835
      ENDIF
      IF (runmode .EQ. 'SIMUL') THEN
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = water_supply_to_holyl
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = lokreg2stluc
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = lok2stluc_est
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = lok2stluc_irrig
```

```
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = lokreg2caloos
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = lok2caloos_est
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = lok2caloos_irrig
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = rlokreg2res1
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = rlokreg2res2
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = reg2est1
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = reg2est2
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = holey_out_to_nwc
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = holey_out_to_ds
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = holy_out_to_s140
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = ws_to_lec_s351_nnrc
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = ws_to_lec_s351_hill
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = ws_to_lec_s351_wca3a
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = outflow_from_roten_to_holy
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = sspsfcd_to_sta
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = eswcd_to_sta
JSTR1 = JSTR1 + 1
roten_out_to_wca3anw = outflow_from_roten_to_wca
STRFLW(JSTR1,IDSS) = roten_out_to_wca3anw
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = roten_out_to_s140
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = DIVERS
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = flow_to_s7_sta2byp
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_rec_from_cpbasr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_rec_from_c51asr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_rec_from_hlsbasr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_rec_from_nnrcasr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_rec_from_acmasr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_inj_to_c51asr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_inj_to_hlsbasr
JSTR1 = JSTR1 + 1
STRFLW(JSTR1,IDSS) = vol_inj_to_nnrcasr
```

```
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = vol_inj_to_l31asr
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = WSS151
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = REGS151
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = tcnsrnff(jday) / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = tcrnbsres_to_lok
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = lok2asr
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = asr2lok
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = cresbp2lok / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_to_roten_frm_lok
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = total_reuse
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = emerg_bypass_s7
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = emerg_bypass_s8
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_pipeline_3a
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_pipeline_3a_s150
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_pipeline_2a
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_pipeline_ca1
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_pipeline_ca1_hill + ws_pipeline_ca1_wpb
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s31dsg_to_bay
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = s140fc
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = fplresws(jday) / 1.9835
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = inflow_to_sta6_frm_unit1
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = total_reuse_res
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = cout_to_c51_frm_l8bprs
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = rl8runoff_to_bptres
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = outflow_to_l8s
         DO kk = 1,nsvarea
            jstr1 = jstr1 + 1
            strflw(jstr1,idss) = rloc_seep_for_ws(kk) / 86400.
         END DO
         JSTR1 = JSTR1 + 1
         STRFLW(JSTR1,IDSS) = ws_to_l8_via_s352
```

```
      JSTR1 = JSTR1 + 1
      STRFLW(JSTR1,IDSS) = s331ws
      JSTR1 = JSTR1 + 1
      STRFLW(JSTR1,IDSS) = s331fc
      JSTR1 = JSTR1 + 1
      STRFLW(JSTR1,IDSS) = REGS31
      JSTR1 = JSTR1 + 1
      STRFLW(JSTR1,IDSS) = WSS31
      DO kk = 1,nsvarea
         jstr1 = jstr1 + 1
         strflw(jstr1,idss) = rloc_excess_for_ws(kk) / 86400.
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      DO kk = 1,nsvarea
         jstr1 = jstr1 + 1
         strflw(jstr1,idss) = total_reuse_ws(kk)
      END DO
      JSTR1 = JSTR1 + 1
      STRFLW(JSTR1,IDSS) = s333_reg_lec
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgdmdistapoga(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgrnfistapoga(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgdmdnls(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgrnfnls(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgdmdnels(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = chgrnfnels(jday) / 1.9835
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = uisrnff(jday) / 1.9835

      DO kk = 1,nsvarea
         jstr1 = jstr1 + 1
         strflw(jstr1,idss) = asr_rec_ws(kk)
      END DO
      DO kk = 1,3
         jstr1 = jstr1 + 1
         strflw(jstr1,idss) = asr_rec_ws_cnty(kk)
      END DO
      jstr1 = jstr1 + 1
      strflw(jstr1,idss) = stl_res_backflw
      DO kk = 1,3
         jstr1 = jstr1 + 1
         strflw(jstr1,idss) = asr_rec_vol_wca(kk)
      END DO
      jstr1 = jstr1 + 1
      n_flw_dss_uncond = 147
      strflw(jstr1,idss) = ws_from_acmres_lwdd
      jstr1 = jstr1 + 1
      n_flw_dss_uncond = n_flw_dss_uncond + 1
      strflw(jstr1,idss) = ws_from_acmres_cpbres
      jstr1 = jstr1 + 1
      n_flw_dss_uncond = n_flw_dss_uncond + 1
      strflw(jstr1,idss) = rlok_flow_to_enp
```

```
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = cfsoth1
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = s333_reg_lec_abve_sch
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = s333_reg_lec_bel_sch
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = bflow_s352
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = bflo_via_l8
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = s356_to_cell
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = s356_to_l29
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = rlvseep_to_l30
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = rlvseep_to_cnl(l31nc)
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = rlok_flow_to_enp_s354
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = rlok_flow_to_enp_s150
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = ebwcd_to_sta
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = q715_farms_to_sta
         jstr1 = jstr1 + 1
         n_flw_dss_uncond = n_flw_dss_uncond + 1
         strflw(jstr1,idss) = rlokreg_to_l8
       ENDIF
c
         IF (n_str_if_cond(1) .ne. n_flw_dss_uncond .and. runmode .eq.
     $   'SIMUL') THEN
              WRITE(*,2000) n_str_if_cond(1),n_flw_dss_uncond
 2000         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH) UNCONDITIONALLY NOT MATCHING NO OF VALUES IN CODE. NO OF NAMES
     $OUTPUT = ',I3,'NO OF DATA VALUES IN CODE = ',I3,'. WILL HAVE MISMA
     $TCH.CHECK INPUT  NEAR BOTTOM OF canal_struc_specs.dat FILE.')
              STOP
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         n_flw_dss_nsta = 0
         IF (NMAREA .GT. 0) THEN
            JSTR1 = JSTR1 + 1
```

```
n_flw_dss_nsta = 1
STRFLW(JSTR1,IDSS) = TWSUPLY
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = ws_from_lo_s6
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = ws_from_lo_s5a
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = RFLS155A
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_des_news8
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_des_news7
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_stor_sta34
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_des_news6
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_stor_sta2
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = sugh_runoff_over_des_strsta6
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_stor_sta6
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = rem_runoff_over_stor_sta5
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = outflow_from_sta6_to_wca
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = outflow_from_sta6_to_semcyp
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = outflow_from_sta6_to_s140
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = outflow_to_canal_roten
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = outflow_to_marsh
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = G136EAA
JSTR1 = JSTR1 + 1
n_flw_dss_nsta = n_flw_dss_nsta + 1
STRFLW(JSTR1,IDSS) = S236SO
JSTR1 = JSTR1 + 1
```

```
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = ws_to_sta5
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = ws_to_sta6
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = ws_to_sta34_frm_lok
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = rmiami_runoff_to_sta
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = rnnrc_runoff_to_sta
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = flow_to_sta2_frm_eaa
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = runoff_to_lec_s8
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = runoff_to_lec_s7
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = runoff_to_lec_s6
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = eaa_runff_s5a_to_sta
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = g136_to_sta34
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = g155_flow_from_3a
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = g155_pos
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = sspsfcd_to_lok
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = eswcd_to_lok
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = q715_farms_to_lok
            JSTR1 = JSTR1 + 1
            n_flw_dss_nsta = n_flw_dss_nsta + 1
            STRFLW(JSTR1,IDSS) = ebwcd_to_lok
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
c       WRITE(63,6262) IFYR+IYEAR-1,MONTH,IDAY,(hydraulic_det_time
c    $      (i),i=1,ntotres)
         ENDIF
         IF (n_str_if_cond(2) .ne. n_flw_dss_nsta) THEN
              WRITE(*,2001) n_str_if_cond(2),n_flw_dss_nsta
```

```
2001         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
    $TH), IF NSTAS GT 0, NOT MATCHING NO OF VALUES. NO OF NAMES OUTPUT
    $= ',I3,' NO OF DATA VALUES = ',I3,'. WILL HAVE MISMATCH.CHECK INPU
    $T NEAR BOTTOM OF canal_struc_specs.dat FILE.')
             STOP
         ENDIF
         n_flw_dss_import = 0
         IF (nmarea .GT. 0 .OR. make_up_water_opt .EQ. 'MAKEUP ') THEN

            JSTR1 = JSTR1 + 1
            n_flw_dss_import = 1
            STRFLW(JSTR1,IDSS) = flow_to_import
            JSTR1 = JSTR1 + 1
            n_flw_dss_import = n_flw_dss_import + 1
            STRFLW(JSTR1,IDSS) = flow_to_import_miami
            JSTR1 = JSTR1 + 1
            n_flw_dss_import = n_flw_dss_import + 1
            STRFLW(JSTR1,IDSS) = flow_to_import_nnrc
            JSTR1 = JSTR1 + 1
            n_flw_dss_import = n_flw_dss_import + 1
            STRFLW(JSTR1,IDSS) = flow_to_import_hill
            JSTR1 = JSTR1 + 1
            n_flw_dss_import = n_flw_dss_import + 1
            STRFLW(JSTR1,IDSS) = flow_to_import_wpb
         ENDIF
         IF (n_str_if_cond(3) .ne. n_flw_dss_import) THEN
              WRITE(*,2002) n_str_if_cond(3),n_flw_dss_import
2002          FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
    $TH),IF NSTAS GT 0 OR BMP MAKEUP RULE IN EFFECT,NOT MATCHING NO OF
    $VALUES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = '
    $,I3,'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_spec
    $s.dat FILE.')
             STOP
         ENDIF
         n_flw_dss_sta34 = 0
         IF (istaopt(is8bsn-6) .EQ. 1 .OR. istaopt(is7bsn-6) .EQ. 1)THEN
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = 1
            STRFLW(JSTR1,IDSS) = sta34_out_to_wca3ane
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = sta34_out_to_wca3ane_adj
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = sta34_out_to_wca3anw
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = sta34_out_to_s140
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = emerg_bypass_sta34
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = dischg_to_s8_frm_sta34_adj
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = dischg_to_s7_frm_sta34_adj
```

```
            JSTR1 = JSTR1 + 1
            n_flw_dss_sta34 = n_flw_dss_sta34 + 1
            STRFLW(JSTR1,IDSS) = total_flow_g404
         ENDIF
         IF (n_str_if_cond(4) .ne. n_flw_dss_sta34) THEN
              WRITE(*,2003) n_str_if_cond(4),n_flw_dss_sta34
 2003         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),IF WATER IS ROUTED TO STA34,NOT MATCHING NO OF VALUES. NO OF
     $NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3,
     $'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.da
     $t FILE.')
              STOP
         ENDIF
         n_flw_dss_l8_no_s316 = 0
         IF ( .NOT. opt_l8_prop .AND. runmode .EQ. 'SIMUL') THEN
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_no_s316 = 1
            STRFLW(JSTR1,IDSS) = wsdel_to_l8
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_no_s316 = n_flw_dss_l8_no_s316 + 1
            STRFLW(JSTR1,IDSS) = total_add_supply_to_lwdd
         ENDIF
         IF (n_str_if_cond(5) .ne. n_flw_dss_l8_no_s316) THEN
              WRITE(*,2004) n_str_if_cond(5),n_flw_dss_l8_no_s316
 2004         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),IF HAVE NO MODIFICATIONS IN L8 BASIN,NOT MATCHING NO OF VALUES
     $ NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3,
     $'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.da
     $t FILE.')
              STOP
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         n_flw_dss_l8_w_s316 = 0
         IF (opt_l8_prop) THEN
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_w_s316 = 1
            STRFLW(JSTR1,IDSS) = WSDELFRWPB
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_w_s316 = n_flw_dss_l8_w_s316 + 1
            STRFLW(JSTR1,IDSS) = COUTC10A
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_w_s316 = n_flw_dss_l8_w_s316 + 1
            STRFLW(JSTR1,IDSS) = total_add_supply_to_lwdd
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_w_s316 = n_flw_dss_l8_w_s316 + 1
            STRFLW(JSTR1,IDSS) = wsdel_to_l8s
            JSTR1 = JSTR1 + 1
            n_flw_dss_l8_w_s316 = n_flw_dss_l8_w_s316 + 1
            STRFLW(JSTR1,IDSS) = wsdel_to_c51w
         ENDIF
         IF (n_str_if_cond(6) .ne. n_flw_dss_l8_w_s316) THEN
              WRITE(*,2005) n_str_if_cond(6),n_flw_dss_l8_w_s316
 2005         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),IF HAVE MODIFICATIONS IN L8 BASIN,NOT MATCHING NO OF VALUES. N
     $O OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3,
     $'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.da
     $t FILE.')
```

```
            STOP
        ENDIF
        n_flw_dss_rfops = 0
        IF (type_flow_across_ttrail .NE. 'MINDEL' .and. runmode .eq.
    $   'SIMUL') THEN
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = 1
           STRFLW(JSTR1,IDSS) = s12_reg_total
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = n_flw_dss_rfops + 1
           STRFLW(JSTR1,IDSS) = s12_env_total
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = n_flw_dss_rfops + 1
           STRFLW(JSTR1,IDSS) = s333_reg
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = n_flw_dss_rfops + 1
           STRFLW(JSTR1,IDSS) = s333_env
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = n_flw_dss_rfops + 1
           STRFLW(JSTR1,IDSS) = s355_reg
           JSTR1 = JSTR1 + 1
           n_flw_dss_rfops = n_flw_dss_rfops + 1
           STRFLW(JSTR1,IDSS) = s355_env
        ENDIF
        IF (n_str_if_cond(7) .ne. n_flw_dss_rfops) THEN
             WRITE(*,2006) n_str_if_cond(7),n_flw_dss_rfops
2006         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
    $TH),IF HAVE MIN DEL SCHED OR RF OPS FOR SRS IN ENP,NOT MATCHING NO
    $ OF VALUES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3,
    $'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.da
    $t FILE.')
             STOP
        ENDIF

        n_flw_dss_nstor = 0
        IF (inorth_stor_opt .EQ. 1) THEN
           JSTR1 = JSTR1 + 1
           n_flw_dss_nstor = 1
           STRFLW(JSTR1,IDSS) = rlok_to_north_stor
           JSTR1 = JSTR1 + 1
           n_flw_dss_nstor = n_flw_dss_nstor + 1
           STRFLW(JSTR1,IDSS) = rnorth_stor_to_lok
        ENDIF
        IF (n_str_if_cond(8) .ne. n_flw_dss_nstor) THEN
             WRITE(*,2008) n_str_if_cond(8),n_flw_dss_nstor
2008         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
    $TH),CONCERNING NORTH STORAGE(inorth_stor_opt=1 OR 0),NOT MATCHING
    $NO OF VALUES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3
    $,'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.d
    $at FILE.')
             STOP
        ENDIF

        n_flw_dss_semcyp = 0
        IF (isem_flg .EQ. 1) THEN
           JSTR1 = JSTR1 + 1
           n_flw_dss_semcyp = 1
```

```
            STRFLW(JSTR1,IDSS) = ws_frm_lok_to_semcyp
c            JSTR1 = JSTR1 + 1
c            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
c            STRFLW(JSTR1,IDSS) = ws_frm_lok_to_sugh
            JSTR1 = JSTR1 + 1
            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
            STRFLW(JSTR1,IDSS) = outflow_from_roten_thru_s8_pres
            JSTR1 = JSTR1 + 1
            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
            STRFLW(JSTR1,IDSS) = outflow_from_roten_to_west
            JSTR1 = JSTR1 + 1
            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
            STRFLW(JSTR1,IDSS) = outflow_from_roten_to_wca
            JSTR1 = JSTR1 + 1
            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
            STRFLW(JSTR1,IDSS) = outflow_from_roten_to_semcyp
            JSTR1 = JSTR1 + 1
            n_flw_dss_semcyp = n_flw_dss_semcyp + 1
            STRFLW(JSTR1,IDSS) = sem_dmnd_met_bef_lok / 1.9835
         ENDIF
         IF (n_str_if_cond(9) .ne. n_flw_dss_semcyp) THEN
              WRITE(*,2007) n_str_if_cond(9),n_flw_dss_semcyp
 2007         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),FOR MEETING SEMCYP DEMANDS(isem_flg = 1 OR 0),NOT MATCHING NO
     $ OF VALUES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3,
     $'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.da
     $t FILE.')
              STOP
         ENDIF

         n_flw_dss_loksplit = 0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         IF (lok_split_option .EQ. 1) THEN
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = 1
            STRFLW(JSTR1,IDSS) = s354_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = s351_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = s352_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = dmnds4_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = dmnds236_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = dmndsemind_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = rlokreswsroten
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = flpresws_res
```

```
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = r298dws_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = l8ws_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = wsdelfrwpb_res
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = eallokres1 / 1.9835
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = eallokres2 / 1.9835
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = c43dmd_met_by_lokres / 1.9835
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = c44dmd_met_by_lokres / 1.9835
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = outflow_to_lokres
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = cfsoth_litz1
            JSTR1 = JSTR1 + 1
            n_flw_dss_loksplit = n_flw_dss_loksplit + 1
            STRFLW(JSTR1,IDSS) = cfsoth_res1
         ENDIF
         IF (n_str_if_cond(10) .ne. n_flw_dss_loksplit) THEN
              WRITE(*,2009) n_str_if_cond(10),n_flw_dss_loksplit
 2009         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),CONCERNING LOK SPLIT(lok_split_option = 1 OR 0),NOT MATCHING N
     $O OF VALUES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3
     $,'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.d
     $at FILE.')
              STOP
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
         DO  IA = 1,NSTARFF
            JSTR1 = JSTR1 + 1
            STRFLW(JSTR1,IDSS) = QSTAIN(ISTARF(IA))
         END DO
         if (n_sta_inf_dss .ne. NSTARFF) THEN
              WRITE(*,2010) n_sta_inf_dss,NSTARFF
 2010         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),CONCERNING STA INFLOW(STAs PART OF ECP),NOT MATCHING NO OF VAL
     $UES. NO OF NAMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3
     $,'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.d
     $at FILE.')
              STOP
         ENDIF
C
         n_flw_dss_asreaa = 0
```

```
         IF (iasr_opt_eaa(is8bsn - 6).EQ. 1.OR. iasr_opt_eaa(is7bsn - 6)
     +   .EQ. 1 .OR. iasr_opt_eaa(is5absn-6) .EQ. 1 .OR. nasr_to_eaa
     +   (is8bsn - 6) .GT. 0 .OR. nasr_to_eaa(is7bsn - 6) .GT. 0 .OR.
     +   nasr_to_eaa(is5absn - 6) .GT. 0) THEN
            JSTR1 = JSTR1 + 1
            n_flw_dss_asreaa = 1
            STRFLW(JSTR1,IDSS) = flow_from_asr_for_ws_eaa/1.9835
            JSTR1 = JSTR1 + 1
            n_flw_dss_asreaa = n_flw_dss_asreaa + 1
            STRFLW(JSTR1,IDSS) = flow_from_asr_meet_rem_eaa_dmnd/1.9835
         ENDIF
         if (NADDSTR_ASR_EAA .ne. n_flw_dss_asreaa) THEN
              WRITE(*,2011) NADDSTR_ASR_EAA,n_flw_dss_asreaa
 2011         FORMAT(/,'WARNING: NO OF NAMES OUTPUT TO DSS (B-PART OF PA
     $TH),CONCERNING EAA ASR RECOVERY,NOT MATCHING NO OF VALUES. NO OF N
     $AMES OUTPUT = ',I3,' NO OF DATA VALUES = ',I3
     $,'. WILL HAVE MISMATCH.CHECK INPUT NEAR END OF canal_struc_specs.d
     $at FILE.')
              STOP
         ENDIF
         imore_than_one_eaares_env = 0
c        IF (no_of_res_reg_frm_lok(1) .GT. 1 .OR. no_of_res_reg_frm_lok
c    $   (2) .GT. 1) imore_than_one_eaares_env = 1

         DO ib = 1,neaabsn
            IF (ib .NE. 2) THEN
               IF (iasr_opt_eaa(ib) .EQ. 1 .OR.ires_opt_eaa(ib) .EQ. 1)
     $         THEN
                  jstr1 = jstr1 + 1
                  strflw(jstr1,idss) = flow_to_eaa_res_rnff(ib)
               ENDIF
               DO kk = 1,no_of_res_reg_frm_lok(ib)
                  jstr1 = jstr1 + 1
                  strflw(jstr1,idss) = flow_to_eaa_res_reg(ib,kk)
               END DO
            ELSE
               IF (iasr_opt_eaa(ib) .EQ. 1 .OR.ires_opt_eaa(ib) .EQ. 1
     $         .OR. ires_opt_hill .EQ. 1) THEN
                  jstr1 = jstr1 + 1
                  strflw(jstr1,idss) = flow_to_eaa_res_rnff(ib)
               ENDIF
               DO kk = 1,no_of_res_reg_frm_lok(ib)
                  jstr1 = jstr1 + 1
                  strflw(jstr1,idss) = flow_to_eaa_res_reg(ib,kk)
               END DO
            ENDIF
         END DO
         IF (imore_than_one_eaares_env .GT. 0) THEN
            jstr1 = jstr1 + 1
            strflw(jstr1,idss) = total_env_ws_from_eaares
         ENDIF
         total_res_outflow_to_basin = 0.0
         DO irs = 1,nres_ws
            DO js = 1,noutlets(irs)
               IF (idshg_opt(ires_no(irs),js) .EQ. 1) THEN
                  jstr1 = jstr1 + 1
                  strflw(jstr1,idss) = res_outflow_to_basin(irs,js)
```

```
                  total_res_outflow_to_basin =
     $            total_res_outflow_to_basin + res_outflow_to_basin
     $            (irs,js)
                ENDIF
             END DO
          END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          IF (ires_opt_eaa(is8bsn-6) .EQ. 1 .OR. ires_opt_eaa(is7bsn - 6)
     $   .EQ. 1 .OR. ires_opt_eaa(is5absn-6) .EQ. 1 .OR. ires_opt_hill
     $   .EQ. 1) THEN
             jstr1 = jstr1 + 1
             strflw(jstr1,idss) = total_res_outflow_to_basin
             DO i = 1,neaabsn
                DO k = 1,no_of_res_reg_frm_lok(i)
                   tot_lok_to_eaares = tot_lok_to_eaares + flow_asr_inj
     $             (i,k)
                END DO
             END DO
             JSTR1 = JSTR1 + 1
             STRFLW(JSTR1,IDSS) = tot_lok_to_eaares
          ENDIF
          DO irs = 1,nres_ws
             IF (nasr_from_res(ires_no(irs)) .GT. 0) THEN
                DO is = 1,noutlets(irs)
                   js = istrctr_ws(ires_no(irs),is)
                   IF (idshg_opt(ires_no(irs),is) .NE. 1) THEN
                      IF (inodopt(ires_no(irs),js,3) .EQ. 0) THEN
                         icnl_no = ircnl(ires_no(irs),js,1)
                         char_id_cnl_ws_resasr = cnm(icnl_no)
                         IF (char_id_cnl_ws_resasr(1:3) .ne. 'LWD') THEN
                          IF (nasr_to_cnl(icnl_no) .GT. 0) THEN
                            jstr1 = jstr1 + 1
                            STRFLW(jstr1,idss) = flow_from_asr_ws(irs,is)
     $                      / 1.9835
                          ENDIF
                         ENDIF
                      ENDIF
                   ENDIF
                END DO
             ENDIF
          END DO
c
c  output levee seepage diversion to str2x2.dss
c
       ll = 0
       DO lv = 1,nlevs
          DO i = 1,LVNODES(LV)
             ll = ll + 1
             IF (nint(lvseep_pump_cap(ll)) .GT. 0) THEN
                jstr1 = jstr1 + 1
                strflw(jstr1,idss) = lvseep_divers(ll)
             ENDIF
          END DO
       END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
       IF (iconvey_canal_no(is8bsn - 6) .NE. 0) THEN
```

```
         IF (nint(ws_to_lec_s354) .GT.  0.OR. nint(ws_to_lec_s351_wca3a)
     $   .GT. 0) THEN
            index_ws_from_lok(iwca3a,iconvey_canal_no(is8bsn-6)) = 1
         ELSE
            index_ws_from_lok(iwca3a,iconvey_canal_no(is8bsn-6)) = 0
         ENDIF
      ENDIF
      IF (iconvey_canal_no(is7bsn - 6) .NE. 0) THEN
         IF (nint(ws_to_lec_s351_nnrc) .GT. 0) THEN
            index_ws_from_lok(iwca2a,iconvey_canal_no(is7bsn - 6)) = 1
         ELSE
            index_ws_from_lok(iwca2a,iconvey_canal_no(is7bsn - 6)) = 0
         ENDIF
      ENDIF
      IF (iconvey_canal_no(is6bsn-6) .NE. 0) THEN
         IF (nint(ws_to_lec_s351_hill) .GT. 0 .OR. nint(ws_to_lec_s352)
     $   .GT. 0) THEN
            index_ws_from_lok(iwca1,iconvey_canal_no(is6bsn - 6)) = 1
         ELSE
            index_ws_from_lok(iwca1,iconvey_canal_no(is6bsn - 6)) = 0
         ENDIF
      ENDIF
      nvariab1 = 5 + no_of_subareas_floor(iwca1)
      nvariab2 = 4 + no_of_subareas_floor(iwca2a)
      nvariab3 = 4 + no_of_subareas_floor(iwca3a)
      IF (iconvey_canal_no(is6bsn - 6) .NE. 0) THEN
         index_ws_frm_outside_src_canal_1
     $   =index_ws_frm_outside_src_canal (iwca1,iconvey_canal_no
     $   (is6bsn - 6))
         lec_dmnd_on_lok_1 = lec_dmnd_on_lok(iwca1, iconvey_canal_no
     $   (is6bsn - 6))
         index_ws_from_lok_1 = index_ws_from_lok(iwca1, iconvey_canal_no
     $   (is6bsn - 6))
      ELSE
         index_ws_frm_outside_src_canal_1 = 0
         lec_dmnd_on_lok_1 = 0
         index_ws_from_lok_1 = 0
         lok_criteria_wca1_lok_ws = 0
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (iconvey_canal_no(is7bsn - 6) .NE. 0) THEN
         index_ws_frm_outside_src_canal_2
     $   =index_ws_frm_outside_src_canal (iwca2a,iconvey_canal_no
     $   (is7bsn - 6))
         lec_dmnd_on_lok_2 = lec_dmnd_on_lok(iwca2a, iconvey_canal_no
     $   (is7bsn - 6))
         index_ws_from_lok_2 = index_ws_from_lok(iwca2a,
     $   iconvey_canal_no(is7bsn - 6))
      ELSE
         index_ws_frm_outside_src_canal_2 = 0
         lec_dmnd_on_lok_2 = 0
         index_ws_from_lok_2 = 0
      ENDIF
      IF (iconvey_canal_no(is8bsn - 6) .NE. 0) THEN
         index_ws_frm_outside_src_canal_3
     $   =index_ws_frm_outside_src_canal (iwca3a,iconvey_canal_no
     $   (is8bsn - 6))
```

```
         lec_dmnd_on_lok_3 = lec_dmnd_on_lok(iwca3a, iconvey_canal_no
     $   (is8bsn - 6))
         index_ws_from_lok_3 = index_ws_from_lok(iwca3a,
     $   iconvey_canal_no(is8bsn - 6))
      ELSE
         index_ws_frm_outside_src_canal_3 = 0
         lec_dmnd_on_lok_3 = 0
         index_ws_from_lok_3 = 0
      ENDIF
      IF (runmode .EQ. 'SIMUL') THEN

         WRITE (106,'(i5,2i3,10x,50i3)') ifyr+iyear-1,month,iday,
     $   lok_criteria_wca1_lok_ws,
     $   index_ws_frm_outside_src_canal_1 ,(index_lok_deliv(iwca1,i),i
     $   =1,no_of_subareas_floor(iwca1)) ,logic_floor(iwca1),
     $   lec_dmnd_on_lok_1 ,index_ws_from_lok_1,
     $   index_ws_frm_outside_src_canal_2 ,(index_lok_deliv(iwca2a,i),i
     $   =1, no_of_subareas_floor(iwca2a)),logic_floor(iwca2a) ,
     $   lec_dmnd_on_lok_2,index_ws_from_lok_2 ,
     $   index_ws_frm_outside_src_canal_3 ,(index_lok_deliv(iwca3a,i),i
     $   =1,no_of_subareas_floor(iwca3a)) ,logic_floor(iwca3a),
     $   lec_dmnd_on_lok_3,index_ws_from_lok_3

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      ENDIF
c
      jstrc = jstrc + 1
      cnlstage(jstrc,idss) = stagelo
c
      RETURN
      END
```

ssm.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: ssm.F,v 1.8 2003/11/17 19:41:25 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ssm.F,v $
C   $Name: rel-5-5 $
C
C
C     Given lake okeechobee stage, storage and date this function
C     computes water supply releases to the lake service area. this
C     plan takes effect when the lake stage is projected to fall
C     below 11' nvgd during the dry season. The function is based
C     on the documentation on the Lake Okeechobee Supply-Side
C     Management Plan (Alan Hall, 1991). The code was written by
C     Paul Trimble and Lehar Brion.
C
C     required input:  1. stglake - lake stage (ft.,ngvd)
C                      2. stolake - lake storage (af)
C                      3. mon     - month
C                      4. iday    - day
C                      5. dmd     - lake service area demand (af/day)
C                      6. dellec  - delivery to lec (af/day)
C                      7. sch     - schedule below which supply-side
C                                   management is imposed
C
C     output:          1. ssmlake - actual delivery or lake service
C                                   area demand met
C
C---  dimension and initialize projected et, rf, and normal water use
C---  for the remaining dry season months of october thru may. daymon
C---  defines the number of days in a month. bor defines water borrowed
C---  from a given month.
C
C-----------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

```
      SUBROUTINE ssm (stglake, curryear, mon, iday, dmd, dellec, sch,
     & ssmlake, lok_targ_level, iday_lok_in_ssm, lok_split_option,
     & fract_rain_litzone, wup0, wup, lok_targ_level_wet1,
     & lok_targ_level_wet2, ibm_initc, ifreq_init,
     & initialize_annually_opt, iyear, day1, dryseas, nwgoalsto,
     & mon_targ_wet, iday_targ_wet, ifyr, jday, etp, rfp, iphase)

      INCLUDE 'wmm_par.inc'

      INTEGER curryear,daymon(12), iday, iday_targ_wet(max_n_goal_ssm),
     & mon, mon_targ_wet(max_n_goal_ssm)

      LOGICAL day1, dryseas

      REAL bor(12), borwk(12), credit(12), dellec, dmd, etp(12),
     & etp_red(12), lok_targ_level, lok_targ_level_wet1(max_n_goal_ssm),
     & lok_targ_level_wet2(max_n_goal_ssm), rfp(12), rfp_red(12), sch,
     & stglake, wup(12), wup0(12)

      DATA daymon /31,28,31,30,31,30,31,31,30,31,30,31/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      daymon(2) = 28
      IF (mod(curryear,4) .EQ. 0) daymon(2) = 29
      gtotssm = 0.0
      gtotdmd = 0.0
      DO m = 1,12
         rfp_red(m) = rfp(m)
      END DO
      IF (lok_split_option .EQ. 1) THEN
         DO m = 1,12
            rfp_red(m) = rfp(m)
            etp_red(m) = etp(m)
            etp(m) = etp_red(m)
         END DO
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c***********************************************************************
c
c     IF not in dry season, DO wet season ssm, IF necessary
c
c***********************************************************************
c
c
c---  initialize variable used to determine phase restriction
c---  based on 40E-21.231
c
         projected_sto = stg2sto(stglake,0)
         et_rf_adjust  = 0.0
         remuse_adjust = 0.0
c
      IF (mon .EQ. 5) THEN
         etp1 = 0.0
         rfp1 = 0.0
         wup1 = 0.0
```

```
      ELSE
         IF (iday .EQ. 1) monext = monchk(mon + 1)
         etp1 = etp(monext)
         rfp1 = rfp_red(monext)
         wup1 = wup(monext)
      ENDIF
c
c adjust the available storage to account for the net effect of et and rf
c
      etmon = etp(mon) - etp1
      rfmon = rfp_red(mon) - rfp1
c
      et_rf_adjust = (etp(mon) - rfp_red(mon)) - (etmon-rfmon) /
     $               daymon(mon)*(iday - 1)
c
c---  calculate normal use for the remainder of the dry season
c
      remuse_adjust = (wup(mon) - (wup(mon) - wup1)/daymon(mon)
     $       * (iday - 1))
c
c---
c
      projected_sto = projected_sto - et_rf_adjust - remuse_adjust
c
c---  compare against end-of-dry-season target storage
c
      if(projected_sto.lt.stg2sto(lok_targ_level,0))then
c phase level is at least 3
         iphase = 3
      else
c phase level is at least 1
         iphase = 1
      endif
      IF ( mon .GE. 6 .AND. mon .LE. 9 ) THEN
c        IF (mon .EQ. 6 .AND. iday .EQ. 1 .AND. stglake.LT.sch)issm = 1
         IF (stglake.LT.sch)issm = 1
         IF (issm .EQ. 1) THEN
            ssmlake = ssmwet(stglake,mon,iday,dmd,dellec,sch ,
     $      lok_split_option , fract_rain_litzone  ,
     $      wup,lok_targ_level_wet1,lok_targ_level_wet2,
     $      ibm_initc,ifreq_init,initialize_annually_opt,
     $      iyear,day1,nwgoalsto,jday,mon_targ_wet,iday_targ_wet,ifyr,
     $      etp,rfp)
            iday_lok_in_ssm = 1
clmb             print *,ssmlake,stglake,dmd
         ELSE
            issm = 0
            iday_lok_in_ssm = 0
            ssmlake = dmd
         ENDIF
         RETURN
      ELSE
         issm = 0
      ENDIF
      IF (lok_split_option .EQ. 0) THEN
         stolake = stg2sto(stglake,0)
      ELSE
```

```
         stolake = stg2stolitz(stglake,0)
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c---  initialize day counter at the beginning of the dry season (oct. 1)
c---  and initially set END of dry season goal storage to the
c---  water level equivalent to 11' ngvd
c
      IF ( mon .EQ. 10 .AND. iday .EQ. 1  .OR. (((mon .EQ. ibm_initc
     &   .and. mod(iyear-1,ifreq_init) .eq. 0 .and.
     &   initialize_annually_opt .eq. 1) .or. DAY1) .and. DRYSEAS)) THEN
c         write(12,250)
c250      format('   10    1 1988',/,
c     $'wuday     avail     remuse    bormax    supply    dmd       actbo
c     $r    surplus   ssmlake   dmet      ')
         idaycnt = 0
         IF (lok_split_option .EQ. 0) THEN
            goalsto = stg2sto(lok_targ_level,0)
            rlok_targ_level1 = lok_targ_level - 1.0
            goalsto1 = stg2sto(rlok_targ_level1,0)
         ELSE
            goalsto = stg2stolitz(lok_targ_level,0)
            rlok_targ_level1 = lok_targ_level - 1.0
            goalsto1 = stg2stolitz(rlok_targ_level1,0)
         ENDIF
         sumlec   = 0.0
         DO 10 i =  1, 12
            bor(i) = 0.0
            borwk(i) = 0.0
            wup(i) = wup0(i)
   10    CONTINUE
      ENDIF
c
      IF (iday .EQ. 1) THEN
         ssmtot=0.0
         dmdtot=0.0
         credit(mon) = 0.0
      ENDIF
c
c---  increment dry season day counter
c
      idaycnt = idaycnt + 1
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c**********************************************************************
c
c     IF beginning of a new week recompute allowable monthly allocations,
c     allowable daily allocations, allowable daily borrowing, and actual
c     daily allocations based on allowable daily allocation, available
c     storage, and remaining normal dry-season water use
c
c**********************************************************************
c
      IF ( mod(idaycnt - 1,7) .NE.  0) go to 60
c
c--- increase allocation for this month (and previous month, IF necessary)
c--- by actual borrowing and decrease allocation for the fourth month
```

```
c--- into the dry-season by the same amount
c
      monprev = mon - 1
      IF (mon.EQ.1) monprev = 12
      IF (mon .NE. 10) THEN
         DO 30 ij = 1,2
            mn = monprev
            IF (ij .EQ. 2) mn = mon
            IF (borwk(mn) .EQ. 0.) GO TO 30
            DO 20 i = 1,4
               iadj = monchk(mn+i)
               wup(iadj) = wup(iadj) - borwk(mn)
   20       CONTINUE
            borwk(mn) = 0.0
   30    CONTINUE
      ELSE
         IF (borwk(10) .NE. 0.)THEN
            DO 40 i = 1,4
               iadj = monchk(10 + i)
               wup(iadj) = wup(iadj) - borwk(10)
   40       CONTINUE
            borwk(10) = 0.0
         ENDIF
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c---  set variables at the beginning of week
c
      monext = monchk(mon + 1)
      m2next = monchk(mon + 2)
      m4next = monchk(mon + 4)
      m5next = monchk(mon + 5)
c
c---  reduce daily allocation for the week based on accumulated
c---  credit earlier this month
c
      icrdt = 0
      IF (credit(mon) .GT. 0.0) THEN
c
c decrease allocation this month IF credit has been accumulated
c last week; credit is applied only to remainder of the month
c
c8-16
         IF (mon .EQ. 5)THEN
            wup1 = 0.0
         ELSE
            wup1 = wup(monext)
         ENDIF
c8-16
         allocprev = (wup(mon)-wup1) / daymon(mon)*(iday-1)
         allocrem = (wup(mon)-wup1) / daymon(mon) *(daymon(mon)-(iday-1)
     $   ) - credit(mon)
         IF (allocrem .LT. 0.0) THEN
            credit(mon) = -allocrem
            allocrem = 0.0
            wudayr = 0.0
         ELSE
```

```
            credit(mon) = 0.0
            wudayr = allocrem/(daymon(mon)-(iday - 1))
         ENDIF
         allocdiff = (wup(mon)-wup1) - (allocprev + allocrem)
c
c increase allocation 4 months into remainder of dry-season by
c amount of credit accumulated in the past weeks of this month
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
         DO 50 i = 1,4
            iadj = monchk(mon+i)
            wup(iadj) = wup(iadj) + allocdiff
   50    CONTINUE
         icrdt = 1
      ENDIF
c
c---  compute allowable daily allocation based on allowable
c---     monthly allocation for the current month
c---  compute allowable daily borrowing based on one-third of allowable
c---     monthly allocation four months into the future
c
      IF ( daymon(mon)-iday .GE. 7 ) THEN
c8-16
         IF (mon .EQ. 4)THEN
            wup1 = wup(monext)
            wup2 = 0.0
            wup10 = wup0(monext)
            wup20 = 0.0
         ELSE IF(mon .EQ. 5)THEN
            wup1 = 0.0
            wup2 = 0.0
            wup10 = 0.0
            wup20 = 0.0
         ELSE
            wup1 = wup(monext)
            wup2 = wup(m2next)
            wup20 = wup0(monext)
            wup20 = wup0(m2next)
         ENDIF
c8-16
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c week falls entirely within a single month
         IF (icrdt .EQ. 0) THEN
            wuday = ( wup(mon)-wup1 ) / daymon(mon)
            wudayr = wuday
         ELSE
            wuday = wudayr
         ENDIF
         wudayx = ( wup0(mon)-wup10 ) / daymon(mon)
         IF (mon .EQ. 1) THEN
            bormax = 1./3.*( wup0(5)/daymon(1) )
         ELSE
            bormax = 1./3.*((wup0(m4next)-wup0(m5next))/daymon(mon)*7.0)
            bormax =  bormax / 7.0
         ENDIF
      ELSE
c week includes portions of two months, use weighted allocations
```

```
         IF (icrdt .EQ. 0) THEN
            wuday = ( wup(mon)-wup1 )/daymon(mon) *( daymon(mon)
     $      - (iday - 1) ) + ( wup1-wup2 ) / daymon(monext) * ( 7. -
     $      (daymon(mon) - (iday - 1)) )
c
            wuday = wuday / 7.0
            wudayr = wuday
         ELSE
            wuday = wudayr*(daymon(mon)-(iday-1))
     $        +  ( wup1-wup2 )/daymon(monext)
     $        *( 7.-(daymon(mon)-(iday-1))  )
            wuday = wuday/7.0

cbw
         ENDIF
         wudayx = ( wup0(mon)-wup10 )/daymon(mon)
     $        *( daymon(mon)-(iday-1) )
     $        + ( wup10-wup20 )/daymon(monext)
     $        *( 7.-(daymon(mon)-(iday-1)) )
         wudayx = wudayx/7.0
         IF (mon.EQ.1) THEN
            bormax = 1. / 3. * ( wup0(5) / daymon(1) * ( daymon(1) -
     $      (iday - 1) ))
            bormax =  bormax / ( daymon(1) - (iday - 1) )
         ELSE
            IF (mon .EQ. 12)THEN
               wupm6 = 0.0
            ELSE
               wupm6 = wup0( monchk(mon  +6) )
            ENDIF
            bormax = 1./3.*( (wup0(m4next)-wup0(m5next)) / daymon(mon) *
     $      (daymon(mon) - (iday - 1)) + (wup0(m5next) - wupm6) / daymon
     $      (monext) * ( 7. -(daymon(mon) - (iday - 1)) ) )
c
            bormax =  bormax / 7.0
         ENDIF
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c---  calculate available storage based on storage above goal storage,
c---  previous week 7-day delivery to lec, and anticipated et + rf for
c---  the remainder of the dry season
c
c lower goal by sum of last week's discharges made to the lec
c
      goalsto = goalsto - sumlec
      goalsto1 = goalsto1 - sumlec
c      print *,sumlec
c
c reset sum of lec delivery for the week to zero
c
      sumlec = 0.0
c
c compute available storage above season END goal storage
c
      avail = stolake - goalsto
      avail1 = stolake - goalsto1
```

```
c8-16
      IF (mon .EQ. 5) THEN
         etp1 = 0.0
         rfp1 = 0.0
         wup1 = 0.0
      ELSE
         etp1 = etp(monext)
         rfp1 = rfp_red(monext)
         wup1 = wup(monext)
      ENDIF
c8-16
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c adjust the available storage to account for the net effect of et and rf
c
      etmon = etp(mon) - etp1
      rfmon = rfp_red(mon) - rfp1
c
      adjust = (etp(mon) - rfp_red(mon)) - (etmon-rfmon) / daymon(mon)
     $*(iday - 1)
      avail = avail - adjust
      avail1 = avail1 - adjust
      IF ( avail .LT. 0.0 ) avail = 0.0
      IF ( avail1 .LT. 0.0 ) avail1 = 0.0
      remain = wup(mon) - (wup(mon)-wup1)/daymon(mon)*(iday - 1)
c
c---  calculate normal use for the remainder of the dry season
c
      remuse = (wup(mon) - (wup(mon) - wup1)/daymon(mon) * (iday - 1))
c
c---  compute actual daily allocation (supply) for this week
c---  calculate supply based on wuday, avail, and remuse
c
      supply = wuday * avail/remuse
      supply1 = wuday * avail1/remuse
      IF ( supply .GT. wuday ) supply = wuday
      IF ( supply1 .GT. wuday ) supply1 = wuday
c
   60 CONTINUE
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c---  sum discharges made to lower east coast every day of this week
c
      sumlec = sumlec + dellec
c
c**********************************************************************
c
c     IF lake stage is greater than ssm schedule RETURN
c
c**********************************************************************
c
      IF ( stglake .GT. sch ) THEN
         ssmlake = dmd
         addcr = 0.0
         actbor = 0.0
         dmet = 1.0
c
```

```
c---  compute possible accumulation of credit
c
         m4next = monchk(mon + 4)
         IF (bor(m4next) .NE. 0.0) THEN
            monext = monchk(mon + 1)
            wuday = ( wup(mon)-wup(monext) ) / daymon(mon)
            supply = amax1(wuday,dmd)
            addcr = amax1(0.,supply - dmd)
            credit(mon) = credit(mon) + addcr
         ELSE
            addcr = 0.0
         ENDIF
         bor(m4next) = bor(m4next) - addcr
         bor(m4next) = amax1(bor(m4next),0.0)
         borwk(mon) = borwk(mon) - addcr
         borwk(mon) = amax1(borwk(mon),0.0)
         IF (bor(m4next) .EQ. 0.0) credit(mon) = 0.0
         ssmtot = ssmtot + ssmlake
         dmdtot = dmdtot + dmd
c
         IF (dmdtot .GT. 0.0) THEN
            dmettot = ssmtot/dmdtot
         ELSE
            dmettot = 1.0
         ENDIF
c
         surplus = addcr
         IF (iday .EQ. daymon(mon)) THEN
            gtotssm = gtotssm + ssmtot
            gtotdmd = gtotdmd + dmdtot
c            print *,'MONTH TOTALS: ',ssmtot,dmdtot,dmettot
c           ssmtot = 0.0
c           dmdtot = 0.0
         ENDIF
         RETURN
c         GO TO 201
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c***********************************************************************
c
c     determine whether adjustments are needed to account for borrowing
c
c     variables used in borrowing scheme
c         1.actbor  - actual water volume borrowed for the day
c         2.bormax  - maximum allowable borrowing volume for each day
c                     of this week
c         3.supply  - actual normal daily allocation for this week
c         4.short   - shortage computed from actual normal daily
c                     allocation (supply) and lake service area
c                     demand (dmd)
c         5.credit  - keeps account of the actual allocation greater
c                     than the demand which can be used to reduce the
c                     allowable daily allocation for the remainder of
c                     the month;
c                     notes:
c                     1. the reduction is done at the beginning of
```

```
c                         each week
c                      2. credit is accumulated only when borrowing
c                         was done earlier this month, and
c                      3. credit cannot be carried over to the next month
c
c*********************************************************************
c
      actbor  = 0.0
      short = 0.0
      surplus = 0.0
      m4next = monchk(mon + 4)
      IF ( mon .GE. 10 .OR. mon .EQ. 1 ) THEN
         IF ( supply .GT. dmd ) THEN
c accumulate credit only IF net borrowing is positive today
            surplus = supply - dmd
            IF (bor(m4next) .NE. 0.0) THEN
               credit(mon) = credit(mon) + surplus
            ELSE
               surplus = 0.0
            ENDIF
            bor(m4next) = bor(m4next) - surplus
            bor(m4next) = amax1(bor(m4next),0.0)
            borwk(mon) = borwk(mon) - surplus
            borwk(mon) = amax1(borwk(mon),0.0)
            IF (bor(m4next) .EQ. 0.0) credit(mon) = 0.0
         ELSE
c borrow allocation during oct-jan from feb-may, IF necessary
            short = dmd - supply
            actbor = amin1(short,bormax)
            bor(m4next) = bor(m4next) + actbor
            borwk(mon) = borwk(mon) + actbor
            credit(mon) = credit(mon) - actbor
            credit(mon) = amax1(credit(mon),0.0)
c        wu10 = wup(10) - wup(11)
c        wu11 = wup(11) - wup(12)
c        wu12 = wup(12) - wup(1)
c        wu1 = wup(1) - wup(2)
c        wu2 = wup(2) - wup(3)
c        wu3 = wup(3) - wup(4)
c        wu4 = wup(4) - wup(5)
c        wu5 = wup(5)
c        wusum = wu10 + wu11 + wu12 + wu1 + wu2 + wu3 + wu4 + wu5
c        wusum1 = 0.0
         ENDIF
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c*********************************************************************
c
c     compute actual daily delivery (i.e. today's water supply release)
c
c     ssmlake is the actual daily delivery or demand met using supply-
c     side management. it is the minimum between actual daily allocation
c     plus actual borrowing (from future allocations) for the day, and
c     the lake service area demand for the day.
c
c*********************************************************************
```

```
c
      ssmlake =  supply + actbor
      IF (dmd .GT. 0.0) THEN
         rspdmd = ssmlake/dmd
      ELSE
         rspdmd = 1.0
      ENDIF
c     IF (rspdmd.LT.0.33) THEN
c use ssmlake based on lower target IF cutback is more than 67%
c        ssmlake =  supply1+actbor
c     ENDIF
c     print *,supply,supply1,actbor,dmd,ssmlake
      ssmtot = ssmtot + ssmlake
      dmdtot = dmdtot + dmd
      IF (dmdtot .GT. 0.0) THEN
         dmettot = ssmtot/dmdtot
      ELSE
         dmettot = 1.0
      ENDIF
      IF (dmd .GT. 0.0) THEN
         dmet = ssmlake/dmd
      ELSE
         dmet = 1.0
      ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (iday .EQ. daymon(mon)) THEN
         gtotssm = gtotssm + ssmtot
         gtotdmd = gtotdmd + dmdtot
c         print *,'MONTH TOTALS: ',ssmtot,dmdtot,dmettot
c        ssmtot = 0.0
c        dmdtot = 0.0
      ENDIF
      RETURN
c      GO TO 201
c999   print *,'TOTALS: ',gtotssm,gtotdmd,gtotssm/gtotdmd
c      STOP
      END
c
```

ssm_cutback.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: ssm_cutback.F,v 1.13 2005/06/10 13:29:21 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ssm_cutback.F,v $
C   $Name: rel-5-5 $
C
C   This subroutine computes water supply releases to Lake Okeechobee
C   Service area according to the Lake Okeechobee Supply-Side Management
C   Plan (Hall,1991). The source of water is Lake Okeechobee.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE ssm_cutback (s235inf, dmndro, dmds236, dmndagl8, dmds4,
     & dmdsemind, chgdmdistapoga, s4runff, chgrnfistapoga, pctmet,
     & env_ws_cutb_ssm_opt, frac_env_met, runmode, use_ssm, agp, month,
     & month_end_credit, stagelo, stagelo_res, ifyr, iyear, iday,
     & dellec, lok_targ_level, lok_split_option, fract_rain_litzone,
     & wup, wup0, supply_excess, demand_s236, demand_l8, demand_caloos,
     & demand_stlucie, demand_s4bsn, demand_semind, nbrkpt_ssm1, dmnd,
     & ro, jday, lok_ssm_cnt, ssm_vol_cutback_thres, tdmdeaa,
     & neaabsn, itypern, ssmlake, dmd,iday_lok_in_ssm, ssmminfrac,
     & lok_targ_level_wet1, lok_targ_level_wet2, s235_capac, s235_frac,
     & issm_cutb_opt_brghton, sum_alloc_ssm, sum_losa_dmnd,
     & sem_cyp_dmnd, ibm_initc, ifreq_init, initialize_annually_opt,
     & day1, dryseas, nwgoalsto, mon_targ_wet, iday_targ_wet, etp, rfp,
     & pdsi, frac_dmnd_met_wt, frac_dmnd_met_wn, nbrkpt_ssmwt,
     & nbrkpt_ssmwn, iregjulwt, iregjulwn, regstgwt, regstgwn,
     & chgdmdnls, chgrnfnls, chgdmdnels, chgrnfnels,istokabdmdnotmet,
     & s4bsn_to_lok,DMDS236_ORIG,DMDS4_ORIG,istokdmdaccum,dmdistok_orig,
     & pot_del_to_eaa_basin,indx_lok_del,dmnd2,add_dmndagl8,
     & pctmet0,iphase1,pcbruleopt,pcbfrac)

      INCLUDE 'wmm_par.inc'
```

```
      DIMENSION agp(max_n_eaa_condt), chgdmdistapoga(366),
     & chgrnfistapoga(366), dmdsemind(366), dmndro(8,2,2,366),
     & iday_targ_wet(max_n_goal_ssm), mon_targ_wet(max_n_goal_ssm),
     & s4runff(366), sem_cyp_dmnd(366), wup(12), wup0(12),
     & chgdmdnls(366), chgrnfnls(366), chgdmdnels(366), chgrnfnels(366),
     & pot_del_to_eaa_basin(max_n_eaa_condt,2),pcbfrac(4)

      CHARACTER env_ws_cutb_ssm_opt*3,runmode*5,use_ssm*3,pcbruleopt*3

      LOGICAL day1, dryseas

      REAL etp(12), iregjulwn(25), iregjulwt(25), lok_targ_level,
     & lok_targ_level_wet1(max_n_goal_ssm), istokabdmdnotmet,
     & lok_targ_level_wet2(max_n_goal_ssm), pdsi(12), regstgwn(25),
     & regstgwt(25), rfp(12)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (runmode .EQ. 'SIMUL') THEN
         if (lok_split_option .eq. 1) THEN
             stg_area_litz = STG2ARLITZ(stagelo)
             stg_area_lok  = STG2AR(stagelo)
             fract_rain_litzone = stg_area_litz/stg_area_lok
         else
             fract_rain_litzone = 1.0
         endif
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c  recompute supplemental EAA demand to be met by LOK which is
c    subject to SSM
c
      TDMDEAA = 0.0
      DO ia = 1,neaabsn-1
         TDMDEAA = TDMDEAA + pot_del_to_eaa_basin(ia,indx_lok_del)
     &     * 1.9835
      END DO
c
c  total supplemental demand in LOSA to be met by Lake Okeechobee
c
      demand_298 = amax1(agp(4),0.0)/43560.
      demand_s236 = AMAX1(DMDS236,0.0)
      demand_l8 = AMAX1(DMNDAGL8,0.0) + add_dmndagl8
      demand_caloos = dmnd
      demand_stlucie = dmnd2
      demand_s4bsn = DMDS4
      demand_semind = dmdsemind(jday)

      DMD = TDMDEAA + demand_298 + demand_s236 + demand_l8 +
     $demand_caloos + demand_stlucie + demand_s4bsn +
     $dmndro(3,2,itypern,jday) + dmndro(6,2,itypern,jday) +
     $dmndro(7,2,itypern,jday) + dmndro(8,2,itypern,jday)

C      WRITE (*,*) "*****************"
C      WRITE (*,*) jday
C      WRITE (*,*) "TDMDEAA=",TDMDEAA,"demand_s236=",demand_s236
C      WRITE (*,*) "demand_l8=",demand_l8,"demand_caloos=",demand_caloos
```

```
C      WRITE (*,*) "demand_stlucie=",demand_stlucie
C      WRITE (*,*) "demand_s4bsn=",demand_s4bsn
C      WRITE (*,*) "dmndro(3,2,itypern,jday)=",dmndro(3,2,itypern,jday)
C      WRITE (*,*) "dmndro(6,2,itypern,jday)=",dmndro(6,2,itypern,jday)
C      WRITE (*,*) "dmndro(7,2,itypern,jday)=",dmndro(7,2,itypern,jday)
C      WRITE (*,*) "dmndro(8,2,itypern,jday)=",dmndro(8,2,itypern,jday)

c
C-----------------------------------------
C  CALCULATE SUPPLY OF WATER FROM LAKE OKEECHOBEE
C  TO MEET DEMANDS IN ITS SERVICE AREA ACCORDING
C  TO SUPPLY SIDE MANAGEMENT
C-----------------------------------------
      IF (use_ssm .EQ. 'YES') THEN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C  NOTE: NEED TO INCORPORATE SCHEDULE FOR SUPPLY SIDE MGMT(SCH)
C
C  DMD NOW JUST INCLUDES EAA AND S4,S236 BASINS. DEMAND FOR
C  REMAINDER OF BASINS WILL BE ADDED
C
         IF (lok_split_option .EQ. 0 .OR. (lok_split_option .EQ. 1 .AND.
     $   stagelo_res .LT. 2.0)) THEN
            SCH = sched(IDAY,MONTH)
            ssm_line_credit_bot = sched1(IDAY,MONTH,nbrkpt_ssm1)
            call ssm(stagelo,iyear+ifyr - 1,month,iday,dmd,dellec,sch,
     $      ssmlake ,lok_targ_level,iday_lok_in_ssm,
     $      lok_split_option ,fract_rain_litzone,wup0,wup,
     $      lok_targ_level_wet1,lok_targ_level_wet2,ibm_initc,
     $      ifreq_init,initialize_annually_opt,iyear,day1,dryseas,
     $      nwgoalsto,mon_targ_wet,iday_targ_wet,ifyr,jday,etp,rfp,
     $      iphase)
            IF (DMD .GT. 0.0) THEN
               PCTMET = ssmlake / DMD
               PCTMET = AMAX1(PCTMET,ssmminfrac)
               PCTMET = AMIN1(PCTMET,1.0)
               pctcb = 1.0 - PCTMET
               pctmet0 = pctmet
               pctcb0 = pctcb
               iphase0 = iphase
               if(pcbruleopt.eq.'YES') then
c print before adjustment
c      print *,iyear+ifyr - 1,month,iday,iphase,pctcb
                 if(iphase.eq.1) then
c check if really in phase 1 or 2 and corresponding cutback limits
                   if(pctcb.ge.pcbfrac(1)) then
c actually in phase 2; need to limit pctcb to 0.30
                     iphase = 2
                     if(pctcb.gt.pcbfrac(2)) then
c assign max cutback of 30% for phase 2
                       pctcb = pcbfrac(2)
                     endif
                   endif
                 else
c check if really in phase 3 or 4 and corresponding cutback limits
                   if(pctcb.gt.pcbfrac(3)) then
c actually in phase 4; need to limit pctcb to 0.60
                     iphase = 4
```

```
                     if(pctcb.gt.pcbfrac(4)) then
c assign max cutback of 60% for phase 4
                       pctcb = pcbfrac(4)
                     endif
                   endif
                 endif
                 pctmet = 1.0 - pctcb
                 pctmet1 = pctmet
                 pctcb1 = pctcb
                 iphase1 = iphase
               else
                 pctcb = 1.0 - pctmet
                 pctmet1 = pctmet
                 pctcb1 = pctcb
                 iphase1 = iphase
               endif
c print after adjustment
c      print *,iyear+ifyr - 1,month,iday,iphase,pctcb
            ELSE
               pctmet0 = 1.0
               pctcb0 = 0.0
               pctmet1 = 1.0
               pctcb1 = 0.0
               PCTMET = 1.0
            ENDIF
            ssmwatch_l = schedwt(IDAY,MONTH,nbrkpt_ssmwt,iregjulwt
     $       ,regstgwt)
            ssmwarn_l  = schedwn(IDAY,MONTH,nbrkpt_ssmwn,iregjulwn
     $       ,regstgwn)
            rlok_stg_dev_ssm = SCH - stagelo
            if (stagelo .ge. SCH) then
              if (rlok_stg_dev_ssm .gt. pdsi(month)) then
                IF (stagelo .lt. ssmwarn_l ) then
                  pctmet = frac_dmnd_met_wn
                 ELSEIF (stagelo .le. ssmwatch_l) then
                  pctmet = frac_dmnd_met_wt
                ELSE
                  pctmet = 1.0
                  pctmet0 = 1.0
                  pctcb0 = 0.0
                  pctmet1 = 1.0
                  pctcb1 = 0.0
                ENDIF
              else
                pctmet = 1.0
                pctmet0 = 1.0
                pctcb0 = 0.0
                pctmet1 = 1.0
                pctcb1 = 0.0
              endif
            endif
            IF (pctmet .LT. 0.994) lok_ssm_cnt = lok_ssm_cnt + 1
            IF (month .GE. 10 .OR. month .LE. 5) THEN
               IF (stagelo .LT. sch) THEN
                  supply_excess = amax1(ssmlake - dmd,0.0)
                  iday_lok_in_ssm = 3
               ELSE
```

```
                  IF (stagelo .lt. ssmwarn_l ) then
                     iday_lok_in_ssm = 2
                  ELSEIF (stagelo .le. ssmwatch_l) then
                     iday_lok_in_ssm = 1
                  ELSE
                     iday_lok_in_ssm = 0
                  ENDIF
                  supply_excess = 0.0
                  iphase1 = 0
               ENDIF
            ELSE
               IF (stagelo .LT. sch) THEN
                  iday_lok_in_ssm = 3
               ELSEIF (stagelo .le. ssmwarn_l) then
                  iday_lok_in_ssm = 2
                  iphase1 = 0
               ELSEIF (stagelo .le. ssmwatch_l) then
                  iday_lok_in_ssm = 1
                  iphase1 = 0
               ELSE
                  iday_lok_in_ssm = 0
                  iphase1 = 0
               ENDIF
               supply_excess = 0.0
            ENDIF
         ELSE
            iday_lok_in_ssm = 0
            PCTMET = 1.0
            ssmlake = dmd
            supply_excess = 0.0
            iphase1 = 0
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
      ELSE
         iday_lok_in_ssm = 0
         PCTMET = 1.0
         ssmlake = dmd
         supply_excess = 0.0
         iphase1 = 0
      ENDIF
c
c  give credit to LOSA during dry season to delay implementation
c   of cutbacks due to SSM policy
c
      IF (runmode .eq. 'SIMUL') THEN
      IF (month .GE. 10 .AND. month .LE. month_END_credit .AND.
     $month_END_credit .GT. 10 .OR. ((month .GE. 10 .OR. month .LE.
     $month_END_credit) .AND. month_END_credit .LE. 10)) THEN

         sum_vol_cutback = sum_vol_cutback + (1.0 - pctmet) * DMD

         IF (sum_vol_cutback .LT. ssm_vol_cutback_thres .AND. stagelo
     $   .GT. ssm_line_credit_bot) THEN
            pctmet1 = 1.0
         ELSE
            pctmet1 = pctmet
```

```
         ENDIF
      ELSE
         pctmet1 = pctmet
         sum_vol_cutback = 0.0
      ENDIF
      pctmet = pctmet1
c
c  calculate allocation and demand for LOSA for days when deliveries
c    are made to EAA
      sum_alloc_ssm = sum_alloc_ssm + ssmlake
      sum_losa_dmnd = sum_losa_dmnd + dmd
      ENDIF
c
c  assume environmental water supply deliveries from LOK
c   to Everglades is subject to same cutbacks as LOSA
c   (according to SSM) IF so choose as indicated in code below
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      IF (env_ws_cutb_ssm_opt .EQ. 'YES') THEN
         frac_env_met = pctmet
      ELSE
         frac_env_met = 1.0
      ENDIF

c  adjust Seminole Breighton reservation according to SSM
      IF (issm_cutb_opt_brghton .EQ. 1) THEN
         dmdsemind(jday) = dmdsemind(jday) * PCTMET
      ENDIF

c  adjust LOK water supply deliveries to other LOSA basins
      chgdmdnls(jday) = dmndro(6,2,itypern,jday)*pctmet
      chgrnfnls(jday) = dmndro(6,1,itypern,jday)
      chgdmdnels(jday) = dmndro(7,2,itypern,jday)*pctmet
      chgrnfnels(jday) = dmndro(7,1,itypern,jday)

c  Logic for lower istokpoga basin demand above G207 and G208
c  (combined capacity of 535.5 ac-ft)

         dmdistok_orig = dmndro(3,2,itypern,jday) +
     $     dmndro(8,2,itypern,jday)
         dmndro(3,2,itypern,jday) = dmndro(3,2,itypern,jday)*pctmet
         dmndro(8,2,itypern,jday) = dmndro(8,2,itypern,jday)*pctmet +
     $     istokabdmdnotmet

         IF ((dmndro(8,1,itypern,jday) - dmndro(8,2,itypern,jday))
     $      .ge. 0.0) THEN
              dmndro(8,1,itypern,jday) = (dmndro(8,1,itypern,jday) -
     $            dmndro(8,2,itypern,jday))
              dmndro(8,2,itypern,jday) = 0.0
         ELSE
              dmndro(8,2,itypern,jday) = (dmndro(8,2,itypern,jday) -
     $            dmndro(8,1,itypern,jday))
              dmndro(8,1,itypern,jday) = 0.0
         ENDIF
         IF ((535.5 - dmdsemind(jday) - dmndro(8,2,itypern,jday))
     $        .gt. 0.0) THEN
            chgdmdistapoga(jday) = dmndro(3,2,itypern,jday) +
```

```
     $        dmndro(8,2,itypern,jday)
            istokabdmdnotmet = 0.0
            istokdmdaccum = 0
         ELSE
            chgdmdistapoga(jday) = dmndro(3,2,itypern,jday) +
     $        (535.5 - dmdsemind(jday))
            istokabdmdnotmet = dmndro(8,2,itypern,jday) - (535.5 -
     $        dmdsemind(jday))
            istokdmdaccum = 1
         ENDIF
         chgrnfistapoga(jday) = dmndro(3,1,itypern,jday) +
     $       dmndro(8,1,itypern,jday)

c  adjust flow to s236 and s4 basins for water supply according
c  to supply side management
      IF (runmode .EQ. 'SIMUL') THEN
         DMDS4_ORIG = DMDS4
         DMDS4 = DMDS4 * PCTMET
         DMDS236_ORIG = demand_s236
         IF (ifix(DMDS236) .GT. 0) THEN
            DMDS236 = DMDS236 * PCTMET
         ENDIF
      ENDIF
      RETURN
      END
```

stas.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: wwilcox $
C     $Id: stas.inc,v 1.7 2005/02/10 15:54:02 wwilcox Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/stas.inc,v $
C       $Name: rel-5-5 $
-
C
C
C-----------------------------------------------------
      common /STAS/ IRESTME(max_n_res),STRCAP(max_n_res),
     &LRESTME(max_n_res),RSIAPM(12),REDBP(2),NINSTR(max_n_res),
     &NOUTSTR(max_n_res),frac_298r_s(5),struc_cap_298d_s(5),
     &NODEPOS_CANLR(max_n_res,max_nostr_res,max_n_out_loc_str),
     &IXLIN(max_n_res,max_nostr_res,max_n_inf_loc_str),
     &IXLOUT(max_n_res,max_nostr_res),sspsfcd_to_lok,
     &IUPSTRM(max_n_res,max_nostr_res,max_n_out_loc_str),
     &IDNSTRM(max_n_res,max_nostr_res),PCTWMA(max_n_res),
     &IXRW(max_n_res,max_nostr_res,max_n_out_loc_str),g372cap,
     &IYRW(max_n_res,max_nostr_res,max_n_out_loc_str),
     &IWMADS(max_n_res,max_nostr_res),IXCNODE(max_n_res),
     &IYCNODE(max_n_res),RESTIME(max_n_res),sspsfcd_to_sta,
     &STGMX(max_n_res),NDSPTS(max_n_res,max_nostr_res),
     &IYLIN(max_n_res,max_nostr_res,max_n_inf_loc_str),
     &IYLOUT(max_n_res,max_nostr_res),DPHMNOT(max_n_res),
     &DPHWS(max_n_res),NSTRSTA,WEIRCFF(max_n_res,max_nostr_res,2),
     &CRSTELEV(max_n_res,max_nostr_res,2),eswcd_to_lok,
     &WREXP(max_n_res,max_nostr_res,2),eswcd_to_sta,
     &IRCNL(max_n_res,max_nostr_res,max_n_out_loc_str),g370cap,
     &node_reserv(max_n_res,max_n_res_grid_loc),QSTAIN(max_n_res),
     &NBSNTSTA,QL201S,IDIRFRLO(max_n_res),q715_farms_to_lok,
     &INDXSTA(max_n_res),NINDPTS(max_n_res,max_nostr_res),
     &NOUTDPTS(max_n_res,max_nostr_res),q715_farms_to_sta,
     &INODOPT(max_n_res,max_nostr_res,3),ebwcd_to_lok,
     &INCNL(max_n_res,max_nostr_res,max_n_inf_loc_str),
     &IOUTCNL(max_n_res,max_nostr_res),INDXAWS(max_n_res),NSTARFF,
     &ISTARF(max_n_res),NMAWS,NRES(max_ncnls,max_nostr_cnl),
     &PCTRES(max_ncnls,max_nostr_cnl,max_n_res_rec_str_flw),
```

```
&IRSRCVR(max_ncnls,max_nostr_cnl,max_n_res_rec_str_flw),
&SFACTOR(max_n_res),RWIDTH(max_n_res),RLENGTH(max_n_res),
&RMXCPO(max_n_res,max_nostr_res),NNODES(max_n_res),NPUMPTORES,
&IBSNRES(max_n_res),NMAREA,iopt_ws_lec_to_sta_wca_abv_flr,
&INGVHDOPT(max_ncnls,max_nostr_cnl,max_n_substr),
&NSTRTORES(max_n_res),NTOTRES,ebwcd_to_sta,
&DWNSTGMX(max_n_res,max_nostr_res,max_n_out_loc_str),
&OUTLIM(max_n_res,max_nostr_res),
&NCOUNT(max_n_str_node_dep),rem_runoff_over_des_news8,
&rem_runoff_over_stor_sta34,iter_asr,rem_runoff_over_des_news7,
&rem_runoff_over_des_news6,conv_asr,rem_runoff_over_stor_sta2,
&rem_runoff_over_des_strsta6,ista1e_node,rem_runoff_over_stor_sta6,
&rem_runoff_over_stor_sta5,flow_s2_tmp,res_area(max_n_res),
&hydraulic_det_time(max_n_res),mean_ponding_depth(max_n_res),
&PUMPCAP_RES(max_n_res),ires_pump(max_n_eaa_condt),
&frac_lec(max_n_eaa_condt),rmax_cap_298d_to_lok(max_n_eaa_condt),
&avail_vol_to_asr(max_n_res,max_n_asr_frm_res),
&idirect(max_n_res,max_n_res_grid_loc),outf_from_c11res_to_c11w,
&ACSTG(max_n_gage_loc_outp,max_n_cell_loc_gage),
&OFFSET_REG_RES(max_n_res,max_nostr_res,max_n_out_loc_str),
&dischg_bld_acmres,avail_vol_locfc,
&outf_seep_loc_to_res_cell(max_ncells),
&outf_fc_frm_res_to_cell(max_ncells),
&sum_chg_depth_levee_spg(max_ncells),TYPE(max_n_res,max_nostr_res),
&chg_depth_levee_spg(max_ncells),ws_from_acmres_cpbres,sfactmin,
&RESNAME(max_n_res),ires_small_sim(max_n_res),
&res_struc_name_spec_code(max_n_str_name_spec),
&name_res_spec_code(max_n_res_name_spec)
 CHARACTER TYPE*7,RESNAME*6,res_struc_name_spec_code*6
*,name_res_spec_code*6,ires_small_sim*3
 real mean_ponding_depth
 LOGICAL LRESTME
```

stat.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: stat.inc,v 1.9 2004/01/20 15:52:29 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/stat.inc,v $
C       $Name: rel-5-5 $
-
C                                                                       -
C                                                                       -
C-----------------------------------------------------
      common /STAT/ HDC(max_ncnls,max_n_seas),CL(max_ncnls),
     &MINX(max_n_rows),FRFP,MXOV(max_n_rows),ELLS(max_ncells),
     &WIDTH(max_ncnls),CHHC(max_ncnls),MAXX(max_n_rows),QMP,
     &OFML(max_n_lu_type,2),SINF(max_ncells),AQDEP(max_ncells),
     &GWDTH(max_ncnls,max_nostr_cnl),NODCR(max_ncnls),
     &ISUM(max_n_rows),S332CP,DX,DY,TKX(max_ncells),
     &CREL(max_ncnls,max_nostr_cnl),INTO(max_ncnls,max_nostr_cnl),
     &LOCW(max_ncells),GDAR,S(max_ncells),MTRC(max_ncnls),
     &MTRP(max_ncnls,20),PET(max_n_lu_type,12),PUMP(max_n_wells,12),DT,
     &DAY1,NFLPTS,NWELL,IYEAR,MAXY,NREG,LP(max_n_lp_flags),
     &iup_canal_ws(max_ncnls),NMTR,NMC,NCH,IFYR,
     &DETEN_DEF(max_n_lu_type),IDFL(max_ncells),LUTYP(max_ncells),
     &NSTART(max_n_serv_area),NSTEP,NOUT(max_ncnls),
     &NODECN(max_n_tot_cnl_loc),DISTNC(3),OFML_fact(max_n_lu_type,2),
     &pump_ind_rss(max_n_wells_rss,12),NLEV,
     &ISTC(max_n_strt_cnls_ws,max_n_serv_area),NSTR,IBSN(max_ncells),
     &NFLO(max_n_struc_flw,3),LCNB(max_ncnls),KFL(max_n_struc_flw,6),
     &SRZ(max_n_lu_type),DRZ(max_n_lu_type),MAXXT,
     &rate_limit(max_no_lvsp),NSF,
     &MAXXPL(max_n_rows),MINXPL(max_n_rows),MAXYPL,MINYPL,
     &KFLO(max_n_struc_flw),NCALCPT,ICALCPT(max_n_str_lok_in_wca),NZERO,
     &NYEAR,AGDMD,NLEVS,LVNODES(max_no_lvsp),LVSP(max_n_lvsp_loc,8),
     &SRATE(max_no_lvsp,3),hlsbreg,sle_dmnd_factor,
     &RMNSTG(max_n_wcas,max_n_conv_cnl_wca),NSTRCTR(max_ncnls),
     &ISTR(max_ncnls,max_nostr_cnl),ISTRMIN(max_ncnls),
     &IDNCNL(max_ncnls,max_nostr_cnl),IETZON(max_ncells),NETZON,
     &nwell_ind_rss,loc_ind_rss(max_n_wells_rss),
     &DETEN(max_ncells),daily_seep_loss_res(max_ncells),
```

```
     &ICNL_NODE_POS(max_n_lvsp_loc,2),PNDDPTH(max_ncnls,5,max_ncnlloc),
     &s8_grav_flow,s8_pump_flow,s7_pump_flow,s7_grav_flow,nnrreg,
     &grf3amx_prescap,grf2amx_prescap,fllks6_prescap,dmnd_lec_sa3,
     &s150_wsdel,ibsn_no_res(max_n_res),srate_frac(max_no_lvsp),
     &IDNCNL_WS(max_ncnls,max_nostr_cnl,5),carfact(max_ncells),
     &rmean_acstg(max_n_gage_loc_outp,max_n_cell_loc_gage),
     &outflow_from_us27s,outflow_from_us27n,
     &cale_dmnd_factor,sumet(max_ncells),
     &rmin_ofml(max_n_lu_type),frac_reg_flow_nesrs_s333,
     &ides_cap_limit(max_ncnls,max_nostr_cnl),seep_loss_res(max_ncells),
     &adjust_for_area(max_ncells),adjust_factr_area(max_n_res),
     &lutyp_res_loc(max_n_res,max_n_res_grid_loc),
     &re_proport_eaa_rnff,TMPNAME(max_ncnls,max_nostr_cnl)
C-----------------
         REAL KFLO,NFLO,nnrreg
         LOGICAL DAY1,LP
         CHARACTER TMPNAME * 6,re_proport_eaa_rnff*5
         DOUBLE PRECISION HDC,RMNSTG
```

stormd.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: stormd.inc,v 1.5 2003/07/03 20:08:06 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/stormd.inc,v $
C       $Name: rel-5-5 $
C
C
C
C-----------------------------------------------------
C
      common /stormd/ nstorm_day(max_n_days_sim),
     &ndays_prestorm_drawdown,struc_name_drawdown(max_n_str_drwdn),
     &rmxstg_drawdown(max_n_str_drwdn),rmnstg_drawdown(max_n_str_drwdn),
     &icanal_no_drawdown(max_n_str_drwdn),
     &istruc_no_drawdown(max_n_str_drwdn),idrwn,
     &RMNSTG2(max_ncnls,max_nostr_cnl,max_n_seas),
     &RMXSTG2(max_ncnls,max_nostr_cnl,max_n_seas),
     &CREL2(max_ncnls,max_nostr_cnl),idrwnres,
     &res_struc_name_drwdown(max_n_str_res_drwdn),
     &istruc_no_res_drwdwn(max_n_str_res_drwdn),
     &CRSTELEV_drwdwn(max_n_str_res_drwdn),
     &stgmx_drwdwn(max_n_str_res_drwdn),
     &ires_no_drawdown(max_n_str_res_drwdn),
     &CRSTELEV2(max_n_res,max_nostr_res),STGMX2(max_n_res),
     &tw_max_struc_inflw_drwdwn(max_n_str_drwdn,max_n_ds_cnl_lim),
     &CNLSTG2(max_ncnls,max_nostr_cnl,max_n_ds_cnl_lim),
     &RMXSTG0(max_ncnls,max_nostr_cnl,max_n_seas),
     &RMNSTG0(max_ncnls,max_nostr_cnl,max_n_seas)
C---------------------------
      character struc_name_drawdown*6,res_struc_name_drwdown*6
```

STRCOUNT.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: STRCOUNT.inc,v 1.4 2003/07/03 20:08:07 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/STRCOUNT.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /STRCOUNT/ istr_index(max_n_str_node_dep)
```

strgrp.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C                                                                       -
C     SCCS ID: %W% %G% SFWMD Planning Department                        -
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: strgrp.inc,v 1.2 2003/07/03 20:08:07 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/strgrp.inc,v $
C       $Name: rel-5-5 $
-
C
C
C-----------------------------------------------------
C Common block for structure flow summary (FLOSUM) subroutine
      COMMON /STRGRP/ MAXDMON, ISUBS, FLODAY(175),FLOMON(175)
C-----------------
```

struc_lok_in_wca_flw.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: struc_lok_in_wca_flw.F,v 1.14 2005/06/10 13:29:21 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/struc_lok_in_wca_flw.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE COMPUTES DISCHARGES FOR STRUCTURES DEPENDENT ON
C   BASIN DEMAND/RUNOFF COMPUTATION OR DEPENDENT ON OPERATIONAL CRITERIA
C   PRIMARILY AS A FUNCTION OF GRID CELL STAGE. STRUCTURES SIMULATED IN
C   THIS ROUTINE ARE EAA STRUCTURES AND OTHERS NORTH OF EVERGLADES.
C
C   ROUTINE WILL RETURN THE KFLO ARRAY CONTAINING DISCHARGES TO KNFLOWS.
C
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE struc_lok_in_wca_flw (crmin, holyinq, pctmet, wsneedl8,
     & jday, divers, runoff_south, runoff_298_districts, runoff_avg,
     & water_supply_to_holyl, ifacteaa, rmonthly_loss_bmp, runoff_s7,
     & runoff_s150, istr_count, capacc10a_bak, s140fc,
     & ws_to_l8_via_s352, pot_del_to_eaa_basin, frac_eaa_dmnd_met,
     & sum_eaa_dmnd_ssm, sum_alocnnr, indx_lok_del, hdc0, iday_of_week,
     & totrun, rsiap, pfrunf, trig_dpth_to_wt, qhlws, total_small_afact,
     & add_dmndagl8,struc_name_meas)

      INCLUDE  'wmm_par.inc'
      INCLUDE  'abc.inc'
      INCLUDE  'agdata.inc'
      INCLUDE  'c1.inc'
      INCLUDE  'c2.inc'
      INCLUDE  'routc.inc'
      INCLUDE  'stat.inc'
      INCLUDE  'wrout.inc'
```

```
      INCLUDE  'stas.inc'
      INCLUDE  'simflows.inc'
      INCLUDE  'wcaot.inc'
      INCLUDE  'lakwca.inc'
      INCLUDE  'environ.inc'
      INCLUDE  'eaa_caps.inc'
      INCLUDE  'eaa_pump_caps.inc'
      INCLUDE  'wcasim.inc'
      INCLUDE  'opersched.inc'
      INCLUDE  'add_misc.inc'

      DIMENSION crmin(max_ncnls,max_n_seas), hdc0(max_ncnls,max_n_seas),
     & pfrunf(max_n_eaa_condt), pot_del_to_eaa_basin(max_n_eaa_condt,2),
     & qhlws(max_n_eaa_condt), runoff_avg(2),
     & runoff_south(max_n_eaa_condt), sum_eaa_dmnd_ssm(max_n_eaa_condt),
     & total_small_afact(max_ncells)

      DOUBLE PRECISION crmin,hdc0,stg_down,hw,tw,ds_head,head_diff
      CHARACTER struc_name_meas(max_n_struc_meas)*6
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IF (.not.opt_l8_prop) THEN
         call match('L8CP  ',1,6,struc_name_meas,nflpts,6,imatch)
         if (imatch .eq. 0) THEN
            hdc(il8,iseas)=min(hdc0(il8,iseas),stagelo+1.0-chdep(il8))
         else
           if (kflo(iflc10a) .lt. -0.0001) then
              hdc(il8,iseas) = min(hdc0(il8,iseas),stagelo+1.0
     $         - chdep(il8))
           else
              hdc(il8,iseas) = hdc0(il8,iseas)
           endif
         endif
      END IF

      wss354=0.0

C  DEFINE THE LOCATIONS OF THE KEY GAGES IN THE WATER CONSERVATION AREAS

      DO i=1,nwcas
        trig_wca_stg(i)=0.0
        rmean_lsel(i)=0.0
        DO j=1,n_gage_loc(i)
          dpthgw=(ells(icell_no_wca_loc(i,j))-h(icell_no_wca_loc(i,j)))*
     &     s(icell_no_wca_loc(i,j))
          IF (pond(icell_no_wca_loc(i,j)).gt.dpthgw) THEN
            wca_stg=pond(icell_no_wca_loc(i,j))-dpthgw+
     &       ells(icell_no_wca_loc(i,j))
          ELSE
            wca_stg=(pond(icell_no_wca_loc(i,j))-dpthgw)/
     &       s(icell_no_wca_loc(i,j))+ells(icell_no_wca_loc(i,j))

          END IF
          trig_wca_stg(i)=trig_wca_stg(i)+wca_stg/n_gage_loc(i)

          rmean_lsel(i)=rmean_lsel(i)+ells(icell_no_wca_loc(i,j))/
     &     n_gage_loc(i)
```

```
        END DO
        IF (i.ne.idn_wca_indx(is8bsn-6)) THEN
          IF (trig_wca_stg(i).lt.rmean_lsel(i)+0.5) THEN
            IF (iconvey_canal_no_wca(i).ne.0) THEN
              trig_wca_stg(i)=chdep(intcnl(i,iconvey_canal_no_wca(i)))
            END IF
          END IF
        END IF
      END DO
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
      IF (runmode.ne.'CALIB') THEN
        kflo(ifls6)=0.
        kflo(ifls150)=0.
      END IF
C
C
      DO 10 ind=1,ncalcpt
        jflo=icalcpt(ind)
        kflo(jflo)=0
   10 CONTINUE
C-----------------------------------------------
C
C
C  route USSG Unit 1 runoff to L-28
C
      IF (istaopt(10).eq.0) THEN
        IF (dryseas) THEN
          frac_runoff_c139=0.05
        ELSE
          frac_runoff_c139=0.10
        END IF
        str_inflow=frac_runoff_c139*(amax1(kflo(ig155),0.0)+kflo(ig89)+
     &   kflo(ig88))
        qu(is140up)=qu(is140up)+str_inflow*86400.
        inflow_to_sta6_frm_unit1=str_inflow
      END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C
C     IF (runmode .EQ. 'CALIB') THEN
C       istr_count = istr_count + nflpts
C     ELSE
C       istr_count = nflpts
C     ENDIF
      DO 50 index=1,ncalcpt
        ncount(ictsta(index))=ncount(ictsta(index))+1
        iflo=icalcpt(index)
C
        istr_count=istr_count+1
        IF (icode(index).eq.'GEN') THEN
C
          IF (ihw_opt(index).eq.1) THEN
            gwdpth=(ells(node_hw(index))-h(node_hw(index)))*
     &       s(node_hw(index))
```

```
            IF (pond(node_hw(index)).ge.gwdpth) THEN
              hw=pond(node_hw(index))-gwdpth+ells(node_hw(index))
            ELSE
              hw=(pond(node_hw(index))-gwdpth)/s(node_hw(index))+
     &         ells(node_hw(index))
            END IF
            avvol=(hw-ells(node_hw(index))-
     &       deten_def(lutyp(node_hw(index))))*(gdar/86400.)
     &       *(1.0-total_small_afact(node_hw(index))
     &       -carfact(node_hw(index)))
            avvol=amax1(avvol,0.0)
          ELSE
            icanal_up=iup_canal_index(index)
            hw=chdep(icanal_up)+hdc(icanal_up,iseas)
            avvol=(hw-cstg_bkpt(index,1))*cl(icanal_up)*width(icanal_up)
     &       +cgseep(icanal_up,cstg_bkpt(index,1),0,0.)+
     &       tovflc(icanal_up,cstg_bkpt(index,1),0,0.,0,1)+qu(icanal_up)
            avvol=amax1(avvol,0.0)/86400.
c              if (flnm(iflo).eq.'S140  ')
c    +         print *,hw,cstg_bkpt(INDEX,1),QU(icanal_up)/86400.,' KKK'
          END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          stage=0.0
          IF (n_add_grid_loc(index).gt.0) THEN
            DO i=1,n_add_grid_loc(index)
              gwdpth=(ells(node_add(index,i))-h(node_add(index,i)))*
     &         s(node_add(index,i))
              IF (pond(node_add(index,i)).ge.gwdpth) THEN
                stage1=pond(node_add(index,i))-gwdpth+
     &           ells(node_add(index,i))
              ELSE
                stage1=(pond(node_add(index,i))-gwdpth)/
     &           s(node_add(index,i))+ells(node_add(index,i))
              END IF
              stage=stage+stage1/n_add_grid_loc(index)
            END DO
          ELSE
            stage=hw
          END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          IF (itw_opt(index).eq.1) THEN
            gwdpth=(ells(node_tw(index))-h(node_tw(index)))*
     &       s(node_tw(index))
            IF (pond(node_tw(index)).ge.gwdpth) THEN
              tw=pond(node_tw(index))-gwdpth+ells(node_tw(index))
            ELSE
              tw=(pond(node_tw(index))-gwdpth)/s(node_tw(index))+
     &         ells(node_tw(index))
            END IF
          ELSE
            tw=chdep(idn_canal_index(index))+hdc(idn_canal_index(index),
     &       iseas)
          END IF
          ds_head=max(tw,cstg_bkpt(index,1))
          head_diff=max(hw-ds_head,0.0)
```

```
          IF (type_flow_s(index).eq.'GRAV') THEN
            rmax_dschg=dischg_c(index)*head_diff**expon(index)
          ELSE
            rmax_dschg=dischg_c(index)
          END IF
          IF (stage.ge.stg_bkpt(index,1)) THEN
            qcap=rmax_dschg
          ELSE IF (stage.gt.stg_bkpt(index,2)) THEN
            qcap=rmax_dschg/(stg_bkpt(index,1)-stg_bkpt(index,2))*
     &       (stage-stg_bkpt(index,2))
          ELSE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            qcap=0.0
          END IF
          qcapt=qcap
          IF (flnm(iflo).eq.struc_name_spec_code(21)) THEN
            qcapt=qcap+qu(icanal_up)/86400.
            s140fc=amin1(qcapt,avvol,rmax_dschg)-qu(icanal_up)/86400.
            s140fc=amax1(s140fc,0.0)
          END IF
c           print *,QCAPT,QCAP,QU(icanal_up) / 86400.,cnm(icanal_up)
c    +        ,AVVOL,rmax_dschg,'  S140 IN ROUTE  ',QU(icanal_up)/86400.
          cout=amin1(qcapt,avvol,rmax_dschg)
          kflo(iflo)=cout
C
          IF (itw_opt(index).eq.0) THEN
            icnl_dns=idn_canal_index(index)
            dqu(icnl_dns)=dqu(icnl_dns)-cout
            dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
   20       icnl_dns=iup_canal_ws(icnl_dns)
            IF (icnl_dns.le.nch) THEN
              dqu(icnl_dns)=dqu(icnl_dns)-cout
              dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
              GO TO 20
            END IF
          END IF
        ELSE
C
C-----------------------------------------------------------------
C               EAST EVERGLADES RES AREA FLOOD PROTECTION
C-----------------------------------------------------------------
C
C
C23456789*123456789*123456789*123456789*123456789*123456789*123456789*12
          IF (flnm(iflo).eq.struc_name_spec_code(14)) THEN
C
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            stgres=0.0
            DO i=1,n_add_grid_loc(index)
              gwdpth=(ells(node_add(index,i))-h(node_add(index,i)))*
     &         s(node_add(index,i))
              IF (pond(node_add(index,i)).ge.gwdpth) THEN
                stgres1=pond(node_add(index,i))-gwdpth+
     &           ells(node_add(index,i))
              ELSE
                stgres1=(pond(node_add(index,i))-gwdpth)/
```

```
     &           s(node_add(index,i))+ells(node_add(index,i))
               END IF
               stgres=stgres+stgres1/n_add_grid_loc(index)
             END DO
             dcoeff_c=dischg_c(index)/(stg_bkpt(index,1)-stg_bkpt(index,
     &        3))
             iresid=iup_canal_index(index)
C
             IF (stgres.gt.stg_bkpt(index,2).or.lreson.or.chdep(iresid)
     &        .ge.cstg_bkpt(index,1)) THEN
               lreson=.true.
               IF (chdep(iresid).gt.cstg_bkpt(index,2)) THEN
                 IF (stgres.ge.stg_bkpt(index,1)) THEN
                   respmp=dischg_c(index)
                 ELSE
                   respmp=dcoeff_c*(stgres-stg_bkpt(index,3))
                 END IF
               ELSE
                 respmp=0.0
               END IF
               IF (stgres.lt.stg_bkpt(index,3)) THEN
                 lreson=.false.
                 respmp=0.0
               END IF
             ELSE IF (stgres.gt.stg_bkpt(index,3).and.lreson) THEN
               respmp=dcoeff_c*(stgres-stg_bkpt(index,3))
             ELSE
               respmp=0.0
               lreson=.false.
             END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
             avolres=(chdep(iresid)-crmin(iresid,iseas))*cl(iresid)*
     &        width(iresid)+cgseep(iresid,crmin(iresid,iseas),0,0.)+
     &        tovflc(iresid,crmin(iresid,iseas),0,0.,0,1)
             avolres=amax1(avolres,0.0)/86400.
             respmp=amin1(avolres,respmp)
C
             IF (itw_opt(index).eq.0) THEN
               icnl_dns=idn_canal_index(index)
               dqu(icnl_dns)=dqu(icnl_dns)-respmp
               dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
   30          icnl_dns=iup_canal_ws(icnl_dns)
               IF (icnl_dns.le.nch) THEN
                 dqu(icnl_dns)=dqu(icnl_dns)-respmp
                 dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
                 GO TO 30
               END IF
             END IF
             kflo(iflo)=respmp
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------------------------------------------------------
C              CULVERT FLOW FROM WPB CATCHMENT AREA
C-----------------------------------------------------------------
           ELSE IF (flnm(iflo).eq.struc_name_spec_code(13)) THEN
             nodcat=node_hw(index)
             tw=chdep(idn_canal_index(index))+hdc(idn_canal_index(index),
```

```
     &       iseas)
             gwdpth=(ells(nodcat)-h(nodcat))*s(nodcat)
             IF (pond(nodcat).ge.gwdpth) THEN
               stage=pond(nodcat)-(ells(nodcat)-h(nodcat))*s(nodcat)+
     &          ells(nodcat)
             ELSE
               stage=(pond(nodcat)-gwdpth)/s(nodcat)+ells(nodcat)

             END IF
             avvol=(stage-ells(nodcat)-deten_def(lutyp(nodcat)))*(gdar/
     &       86400.)*(1.0-carfact(nodcat)-total_small_afact(nodcat))
             avvol=amax1(avvol,0.0)
             IF (loxenv.eq.'FALSE') THEN
               IF (stage.gt.tw) THEN
                 cout=amin1(avvol,dischg_c(index))
               ELSE
                 cout=0.0
               END IF
             ELSE
               IF (lwslo(iloxah)) THEN
                 frac_dischg=(stage_import_wca(iloxah)-
     &            avg_sim_wca_stg(iloxah))/0.20
                 frac_dischg=amin1(frac_dischg,1.0)
                 cout=amin1(dischg_c(index),avvol)*frac_dischg
               ELSE
                 cout=0.0
               END IF
             END IF
C     PRINT *,LWSLO(iloxah),avg_sim_wca_stg(iloxah)
C    +, stage_import_wca(iloxah),COUT,'  LOXAH',POND(LOXNOD1)
C    +, ELLS(LOXNOD1),H(LOXNOD1),RAIN(LOXNOD1),RAIN(LOXNOD2)
C    +, H(NODCAT),POND(NODCAT)
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
             kflo(iflo)=cout
C
C-----------------------------------------------------------------
C      *******    EVERGLADES AGRICULTURAL AREA     *********
C-----------------------------------------------------------------
C
C-----------------------------------------------------------------
C                LOCAL AGRIC ALONG SOUTHERN RIM OF LAKE
C                         OKEECHOBEE
C-----------------------------------------------------------------
           ELSE IF (flnm(iflo).eq.struc_name_spec_code(12)) THEN
             capac=dischg_c(index)
             q=agp(ilagbsn-6)/86400.
             IF (q.ge.0.0) THEN
C
               q=agp(ilagbsn-6)/86400.*pctmet
               iagflo(iag(ilagbsn),1)=-nint(agp(ilagbsn-6)/86400.-q)
               kflo(iflagq)=q
               runoff_298_districts=0.0
             ELSE
               q=abs(q)
               runoff_298_districts=q
             ENDIF
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------------------------------------------------------
C                            S-8  AND   S-3HGS-3
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(11)) THEN
            IF (nsindx(is8bsn-6).eq.1) THEN
              IF (agp(is8bsn-6).ge.0.0) THEN
                qr=1-flwsro(is8bsn-6)
              ELSE
                qr=1-flrnro(is8bsn-6)
              END IF
            ELSE
              qr=1.0
            END IF
C
            qhlwsl=qhlws(is8bsn-6)/86400.
C
            q=agp(is8bsn-6)/86400.
            IF (q.ge.0.0) THEN
              q=0.0
              DO iw=1,n_days_week_del_eaa(indx_lok_del)
                IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &           indx_lok_del)) THEN
                 q=pot_del_to_eaa_basin(is8bsn-6,indx_lok_del)*
     &            frac_eaa_dmnd_met
                 iagflo(iag(is8bsn),1)=-
     &            nint(pot_del_to_eaa_basin(is8bsn-6,indx_lok_del)-q)
                 sum_eaa_dmnd_ssm(is8bsn-6)=0.0
                END IF
              END DO
C
              IF (q.ge.s354mx) THEN
                kflo(ifls3)=s354mx
                iagflo(iag(is8bsn),1)=-nint(q-s354mx)+iagflo(iag(is8bsn)
     &           ,1)
              ELSE
                kflo(ifls3)=q
              END IF
C                print *,stagelo,SCHED(IDAY,MONTH),Q,KFLO(ifls3)
C    +            ,'   S354R  ',frac_eaa_dmnd_met,S354MX,iday_of_week
C    +            ,pot_del_to_eaa_basin(is8bsn-6,indx_lok_del)
C    +            ,(idays_wk_del_eaa_ssm_indx(i,indx_lok_del),i=1,
C    +            n_days_week_del_eaa(indx_lok_del)),IAGFLO
C    +            (IAG(IS8BSN),1),AGP(IS8BSN - 6) / 86400.
              IF (kflo(ifls3).le.s354mx) THEN
                rlimit=s354mx-kflo(ifls3)
              ELSE
                rlimit=0.
              END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              rkhls3=amin1(rlimit,qhlwsl)
              kflo(iflo)=outflow_from_roten_thru_s8+
     &         dischg_to_s8_frm_sta34
              IF (iroute_to_miamic_via_g200a.eq.1) kflo(iflo)=kflo(iflo)
     &         +holey_out_to_ds
```

```
C
              runoff_south(is8bsn-6)=0.0
              cnlcap_ws(is8bsn-6)=cnlcap_ws(is8bsn-6)-amin1(kflo(ifls3),
     &         cnlcap_ws(is8bsn-6))
            ELSE
C
              IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
                q=pfrunf(is8bsn-6)*totrun*bmpred
              ELSE
                q=abs(q)*bmpred
              END IF
              q=q-amin1(flow_reduction_bmp(is8bsn-6),q)
C
              crrnff_bp=crrnff(is8bsn-6)*rcpfacteaa(is8bsn-6)
              IF (runoff_avg(is8bsn-6).gt.crrnff_bp) THEN
                rlokn8=(coefr(is8bsn-6,1)*(runoff_avg(is8bsn-6)-
     &           crrnff_bp)+coefr(is8bsn-6,2))*redbp(1)
                rlokn8=amax1(rlokn8,0.0)
              ELSE
                rlokn8=0.0
              END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              q=q*qr
              qs3=amin1(rlokn8,q)
              IF (qs3.gt.0.0) THEN
                IF (qs3.ge.s3max) THEN
                  kflo(ifls3)=-s3max
                ELSE
                  kflo(ifls3)=-qs3
                END IF
              ELSE
                kflo(ifls3)=0.0
              END IF
              qs8s=amax1(qs3-s3max,0.0)
              q=q-qs3+qs8s
              runoff_south(is8bsn-6)=q
C
              IF (istaopt(is8bsn-6).eq.0.and.(iasr_opt_eaa(is8bsn-6)
     &         .ne.1.and.ires_opt_eaa(is8bsn-6).ne.1)) THEN
                q=runoff_south(is8bsn-6)+outflow_from_roten_thru_s8+
     &           dischg_to_s8_frm_sta34
                IF (iroute_to_miamic_via_g200a.eq.1) q=q+
     &           holey_out_to_ds
C
                IF (q.gt.s8max) THEN
                  kflo(iflo)=s8max
                  iagflo(iag(is8bsn),1)=nint(q-s8max)+iagflo(iag(is8bsn)
     &             ,1)
                ELSE
                  kflo(iflo)=q
                END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              END IF
              IF (kflo(ifls3).ge.0.) THEN
                rlimit=s354mx-kflo(ifls3)
                rkhls3=amin1(rlimit,qhlwsl)
              ELSE
```

```
                rkhls3=0.0
              END IF
              IF (istaopt(is8bsn-6).eq.1) THEN
                kflo(ifls8)=kflo(ifls8)+outflow_from_roten_thru_s8+
     &           dischg_to_s8_frm_sta34
                IF (iroute_to_miamic_via_g200a.eq.1) kflo(ifls8)=
     &           kflo(ifls8)+holey_out_to_ds
              END IF
            END IF
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            total_ws_to_holy=rkhls3
            IF (ells(nodehl)-h(nodehl)
     &       .gt.trig_dpth_to_wt.and.holey_min_level_opt.eq.'YES'
     &       .and.stagelo.gt.rsiap.and.ws_to_holy_opt.eq.'BRUSHF') THEN
              DO 40 iy=maxy,42,-1
                DO 40 node=isum(iy)+1,isum(iy+1)
                IF (ibsn(node).eq.96) THEN
                  IF (ells(node)-h(node).gt.trig_dpth_to_wt) THEN
                    dphgw=(ells(node)-h(node)-trig_dpth_to_wt)*s(node)
                    IF (pond(node).le.dphgw) THEN
                      pond(node)=pond(node)+amin1(total_ws_to_holy*
     &                 86400./(gdar*(1.0-carfact(node))),dphgw)
                      total_ws_to_holy=total_ws_to_holy-
     &                 amin1(total_ws_to_holy,dphgw*gdar/86400.
     &                 *(1.0-carfact(node)))
                    END IF
                  END IF
                END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
   40         CONTINUE
              kflo(ifls3)=kflo(ifls3)+rkhls3

              wss354=wss354+rkhls3
              water_supply_to_holyl=rkhls3
              holyinq=holyinq+rkhls3*86400.
              qhlwsm=qhlwsm+rkhls3*1.9835
            END IF
C
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(3)) THEN
C
C-----------------------------------------------------------------
C                            S-7,S-6   AND S-2 HGS-4
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(10)) THEN
            IF (nsindx(is7bsn-6).eq.1) THEN
              IF (agp(is7bsn-6).ge.0.0) THEN
                qr=1-flwsro(is7bsn-6)
              ELSE
                qr=1-flrnro(is7bsn-6)
              END IF
            ELSE
              qr=1.0
```

```
            END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            q=agp(is7bsn-6)/86400.
C>>
            IF (q.ge.0.0) THEN
C
              q=0.0
              DO iw=1,n_days_week_del_eaa(indx_lok_del)
                IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &           indx_lok_del)) THEN
                  q=pot_del_to_eaa_basin(is7bsn-6,indx_lok_del)*
     &             frac_eaa_dmnd_met
                  iagflo(iag(is7bsn),1)=-
     &             nint(pot_del_to_eaa_basin(is7bsn-6,indx_lok_del)-q)
                  sum_eaa_dmnd_ssm(is7bsn-6)=0.0
                  IF (nint(sum_alocnnr).ge.0) THEN
                    sum_alocnnr=0.0
                  END IF
                END IF
              END DO
C
              IF (q.gt.s351mx) THEN
                kflo(ifls2)=s351mx
                iagflo(iag(is7bsn),1)=-nint(q-s351mx)+iagflo(iag(is7bsn)
     &           ,1)
              ELSE
                kflo(ifls2)=q
                kflo(iflo)=0.
                kflo(ifls6)=0.
              END IF
              IF (q.gt.0.0) THEN
                total_desgn_cap_nnrhl=cnlcap_ws(is7bsn-6)+
     &           cnlcap_ws(is6bsn-6)
                IF (nint(total_desgn_cap_nnrhl).gt.0) THEN
                  cnlcap_ws(is7bsn-6)=cnlcap_ws(is7bsn-6)-
     &             (cnlcap_ws(is7bsn-6)/total_desgn_cap_nnrhl)*
     &             kflo(ifls2)
                  cnlcap_ws(is6bsn-6)=cnlcap_ws(is6bsn-6)-
     &             (cnlcap_ws(is6bsn-6)/total_desgn_cap_nnrhl)*
     &             kflo(ifls2)
                  cnlcap_ws(is7bsn-6)=amax1(cnlcap_ws(is7bsn-6),0.0)
                  cnlcap_ws(is6bsn-6)=amax1(cnlcap_ws(is6bsn-6),0.0)
                END IF
              END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
              runoff_south(is7bsn-6)=0.0
              runoff_south(is6bsn-6)=0.0
              runoff_s150=0.0
              runoff_s7=0.0
            ELSE
C
              DO iw=1,n_days_week_del_eaa(indx_lok_del)
                IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &           indx_lok_del)) THEN
                  IF (nint(sum_alocnnr).ge.0) THEN
                    sum_alocnnr=0.0
```

```
                  END IF
                END IF
              END DO
              IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
                totq=(pfrunf(is7bsn-6))*totrun*bmpred

              ELSE
                totq=abs(q)*bmpred
              END IF
              totq=totq-amin1(flow_reduction_bmp(is7bsn-6),totq)
              crrnff_bp=crrnff(is7bsn-6)*rcpfacteaa(is7bsn-6)
              IF (runoff_avg(is7bsn-6).gt.crrnff_bp) THEN
                rloks2=(coefr(is7bsn-6,1)*(runoff_avg(is7bsn-6)-
     &           crrnff_bp)+coefr(is7bsn-6,2))*redbp(2)
                rloks2=amax1(rloks2,0.0)
              ELSE
                rloks2=0.0
              END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              totq=totq*qr
              rloks2=amin1(rloks2,totq)
C
C     TAKE CARE OF RUNOFF THRU S-7 AND S150
C
              qs7=qs7fact*(totq-rloks2)
              IF (rloks2.gt.0.0) THEN
                IF (rloks2.gt.s2max) THEN
                  kflo(ifls2)=-s2max
                  qs7s=(rloks2-s2max)*(1.0-qs6fact)
                  qs6s=(rloks2-s2max)*qs6fact
                ELSE
                  qs7s=0.0
                  qs6s=0.0
                  kflo(ifls2)=-rloks2
                END IF
              ELSE
                kflo(ifls2)=0.0
                qs7s=0.0
                qs6s=0.0
              END IF
              q=qs7+qs7s
              runoff_s7=q
              qs150=qs150ft*(totq-rloks2)
              total_flow_nnr=runoff_s7+qs150
              IF (istaopt(is7bsn-6).eq.0.and.iasr_opt_eaa(is7bsn-6)
     &         .ne.1.and.ires_opt_eaa(is7bsn-6).ne.1) THEN
                cutw_s150=-1.0
                CALL eaa_prof_lu_table (total_flow_nnr, 12.5, cutw_s150,
     &            2, ifacteaa)
                cutw_s150=amax1(cutw_s150,pmp_int(is7bsn-6))
                s150cap_fc=-1.0
                canal_stg_dn_s150 = chdep(ica3)
                CALL eaaspill (s150cap_fc, cutw_s150, canal_stg_dn_s150
     &           , 8, ifacteaa)
                IF (trig_wca_stg(idn_wca_indx(is7bsn-6))
     &           .gt.regstg_wca(idn_wca_indx(is7bsn-6),1)
     &           .or.runoff_s7.ge.s7max) THEN
```

```
                  qs150_fc=amin1(qs150,s150cap_fc)
                ELSE
                  qs150_fc=0.0
                END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                q=runoff_s7+qs150-qs150_fc
                qs150=qs150_fc
                runoff_s7=q
              END IF
              runoff_s150=qs150
              IF (istaopt(is7bsn-6).eq.0.and.iasr_opt_eaa(is7bsn-6)
     &         .ne.1.and.ires_opt_eaa(is7bsn-6).ne.1) THEN
                IF (q.gt.s7max) THEN
                  kflo(iflo)=s7max
                  iagflo(iag(is7bsn),1)=nint(q-s7max)+iagflo(iag(is7bsn)
     &             ,1)
                ELSE
                  kflo(iflo)=q
                END IF
                kflo(ifls150)=qs150
              END IF
              runoff_south(is7bsn-6)=runoff_s7+qs150

C
C      TAKE CARE OF RUNOFF THRU S-6
C
              qs6=qs6fact*(totq-rloks2)
              q=qs6+qs6s
              runoff_south(is6bsn-6)=q
C
              IF (istaopt(is6bsn-6).eq.0.and.iasr_opt_eaa(is7bsn-6)
     &         .ne.1.and.ires_opt_hill.ne.1) THEN
                IF (q.gt.s6max) THEN
                  kflo(ifls6)=s6max
                  iagflo(iag(is7bsn),1)=nint(q-s6max)+iagflo(iag(is7bsn)
     &             ,1)
                ELSE
                  kflo(ifls6)=q
                END IF
              END IF
            END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(2)) THEN
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(8)) THEN
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(9)) THEN
C
C-----------------------------------------------------------------
C                            S-5A, HGS-5, AND BEL5A
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(7)) THEN
            IF (nsindx(is5absn-6).eq.1) THEN
```

```
              IF (agp(is5absn-6).ge.0.0) THEN
                qr=1-flwsro(is5absn-6)
              ELSE
                qr=1-flrnro(is5absn-6)
              END IF
            ELSE
              qr=1.0
            END IF
            q=agp(is5absn-6)/86400.*qr
C>>
            IF (q.ge.0.0) THEN
C
              q=0.0
              DO iw=1,n_days_week_del_eaa(indx_lok_del)
                IF (iday_of_week.eq.idays_wk_del_eaa_ssm_indx(iw,
     &           indx_lok_del)) THEN
                  q=pot_del_to_eaa_basin(is5absn-6,indx_lok_del)*
     &             frac_eaa_dmnd_met
                  iagflo(iag(is5absn),1)=-
     &             nint(pot_del_to_eaa_basin(is5absn-6,indx_lok_del)-q)
                  sum_eaa_dmnd_ssm(is5absn-6)=0.0
                END IF
              END DO
C
              IF (q.gt.s352mx) THEN
                kflo(iflhg5)=s352mx
                iagflo(iag(is5absn),1)=-nint(q-s352mx)+
     &           iagflo(iag(is5absn),1)
              ELSE
                kflo(iflhg5)=q
              END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
              kflo(iflo)=0.
              divers=0.0
              runoff_south(is5absn-6)=0.0
              flow_reduction_bmp(is5absn-6)=diff_runoff_bmp(is5absn-6,
     &         jday)/1.9835+excess_reduction(is5absn-6)
              excess_reduction(is5absn-6)=flow_reduction_bmp(is5absn-6)
              cnlcap_ws(is5absn-6)=cnlcap_ws(is5absn-6)-
     &         amin1(kflo(iflhg5),cnlcap_ws(is5absn-6))
C
            ELSE
C
              IF (re_proport_eaa_rnff.eq.'TRUE ') THEN
                q=pfrunf(is5absn-6)*totrun*qr*bmpred
              ELSE
                q=abs(q)*bmpred
              END IF
              flow_reduction_bmp(is5absn-6)=diff_runoff_bmp(is5absn-6,
     &         jday)/1.9835+excess_reduction(is5absn-6)
              excess_reduction(is5absn-6)=
     &         amax1(flow_reduction_bmp(is5absn-6)-q,0.0)
              rmonthly_loss_bmp=rmonthly_loss_bmp+
     &         amin1(flow_reduction_bmp(is5absn-6),q)*1.9835
              q=q-amin1(flow_reduction_bmp(is5absn-6),q)
C -----------------------------------
C       ROUTE DIVERSION OF MAX 600 CFS TO STA-2 THRU S-6
```

```
C -------------------------------------
              IF (istaopt(is5absn-6).eq.1.and.istaopt(is6bsn-6).eq.1)
     &         THEN
               divers=amin1(q*0.21,600.0)
              ELSE
               divers=0.0
              END IF
              q=q-divers
              runoff_south(is5absn-6)=q
              IF (istaopt(is6bsn-6).eq.0) THEN
               rem_cap_s6=s6max-kflo(ifls6)
               kflo(ifls6)=kflo(ifls6)+amin1(rem_cap_s6,divers)
               excess_divers=amax1(divers-rem_cap_s6,0.0)
               iagflo(iag(is5absn),1)=nint(excess_divers)+
     &          iagflo(iag(is5absn),1)
              END IF
              IF (istaopt(is5absn-6).eq.1) THEN
               qinflw(indxsta(is5absn-6),ncount(indxsta(is5absn-6)))=
     &          0.0
              END IF
C
              kflo(iflhg5)=0.
              IF (iasr_opt_eaa(is5absn-6).ne.1.and.ires_opt_eaa(is5absn-
     &         6).ne.1) THEN
               IF (q.gt.s5amax) THEN
                kflo(iflo)=s5amax*pctwma(is5absn-6)
                q=q-s5amax*pctwma(is5absn-6)
                iagflo(iag(is5absn),1)=nint(q)+iagflo(iag(is5absn),1)
               ELSE
                kflo(iflo)=q
               END IF
               kflo(ifls5a5)=kflo(ifls5a5)+kflo(iflo)
               s5a2_so=s5a2_so+kflo(iflo)
               IF (istaopt(is5absn-6).eq.1) THEN
                eaa_runff_s5a_to_sta=kflo(iflo)
               END IF
              END IF
            END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------------------------------------------------------
C               S-352
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(1)) THEN
C
C  ws from lok to l8 basin via wbp canal
C
            IF (.not.opt_l8_prop) THEN
              wsneed_rem=wsneedl8
              dmndagl8_rem=wsdmndl8_not_met
              rem_cap_hg5=s352mx-kflo(iflhg5)
              rem_cap_cnl=fllkwpb-amax1(kflo(iflhg5),kflo(ifls5a))
              rem_cap_cnl=amax1(rem_cap_cnl,0.0)
              rem_cap_s5a=amax1(s5amax-kflo(ifls5a),0.0)
              IF (stagelo.lt.sched(iday,month)) THEN
               IF (nint(dmndagl8_rem).gt.0) THEN
                wsdel=amin1(wsneedl8,dmndagl8_rem)*pctmet
```

```
                  cout=amin1(wsdel,rem_cap_hg5,rem_cap_cnl,s5a4max,
     &             rem_cap_s5a)
                  wsneedl8=wsneedl8-cout
                ELSE
                  cout=0.0
                END IF
              ELSE
                wsdel=wsneedl8
                cout=amin1(wsdel,rem_cap_hg5,rem_cap_cnl,s5a4max,
     &           rem_cap_s5a)
                wsneedl8=wsneedl8-cout
              END IF
              kflo(iflhg5)=kflo(iflhg5)+cout
              kflo(ifls5a)=kflo(ifls5a)+cout
              kflo(ifll8)=kflo(ifll8)-cout
              s5a3_no=s5a3_no+cout
              ws_to_l8_via_s352=cout
              IF (nint(dmndagl8_rem).gt.0) THEN
                wsdml8_rem=amin1(wsneed_rem,dmndagl8_rem)
                iagflo(iag(il8agbsn),1)=-nint(amax1(wsdml8_rem-cout,0.0)
     &           )
                qu(il8)=qu(il8)-(dmndagl8_rem-abs(iagflo(iag(il8agbsn),
     &           1)))*86400.
              END IF
            END IF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(6)) THEN
C
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(4)) THEN
C
C-----------------------------------------------------------------
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(5)) THEN
C-------------------------------------------------------------
C              G-261 CULVERT
C-------------------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(15)) THEN
            nodg261=node_hw(index)
            gwdpth=(ells(nodg261)-h(nodg261))*s(nodg261)
            IF (pond(nodg261).ge.gwdpth) THEN
              stgg261=pond(nodg261)-gwdpth+ells(nodg261)
            ELSE
              stgg261=(pond(nodg261)-gwdpth)/s(nodg261)+ells(nodg261)
            END IF
            avvol=(stgg261-ells(nodg261)-deten(nodg261))*gdar/86400.
     &      *(1.0-carfact(nodg261)-total_small_afact(nodg261))
            avvol=amax1(avvol,0.0)
            IF (opt_l8_prop) THEN
              hdrop_adj=0.75
```

```
            ELSE
              hdrop_adj=0.0
            END IF
            stg_down=max(cstg_bkpt(index,1),
     &       chdep(idn_canal_index(index))+hdrop_adj)
            IF (stgg261.gt.stg_down) THEN
              capac=dischg_c(index)*(stgg261-stg_down)**expon(index)
            ELSE
              capac=0.0
            END IF
            cout=amin1(capac,avvol)
            dqu(idn_canal_index(index))=dqu(idn_canal_index(index))-
     &       cout
            dqu(idn_canal_index(index))=amax1(dqu(idn_canal_index(index)
     &       ),0.0)
            kflo(iflo)=cout
C-----------------------------------------------------------------
C                 G-263 CULVERT
C-----------------------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(16)) THEN
            nodg263=node_hw(index)
            gwdpth=(ells(nodg263)-h(nodg263))*s(nodg263)
            IF (pond(nodg263).ge.gwdpth) THEN
              stgg263=pond(nodg263)-gwdpth+ells(nodg263)
            ELSE
              stgg263=(pond(nodg263)-gwdpth)/s(nodg263)+ells(nodg263)
            END IF
C
            avvol=(stgg263-ells(nodg263)-deten(nodg263))*gdar/86400.
     &      *(1.0-carfact(nodg263)-total_small_afact(nodg263))
            avvol=amax1(avvol,0.0)
            IF (opt_l8_prop) THEN
              hdrop_adj=0.50
            ELSE
              hdrop_adj=0.0
            END IF
            stg_down=max(cstg_bkpt(index,1),
     &       chdep(idn_canal_index(index))+hdrop_adj)
            IF (stgg263.gt.stg_down) THEN
              capac=dischg_c(index)*(stgg263-stg_down)**expon(index)
            ELSE
              capac=0.0
            END IF
            cout=amin1(capac,avvol)
            dqu(idn_canal_index(index))=dqu(idn_canal_index(index))-
     &       cout
            dqu(idn_canal_index(index))=amax1(dqu(idn_canal_index(index)
     &       ),0.0)
            kflo(iflo)=cout
C-----------------------------------------------------------------
C                 G-262 CULVERT
C-----------------------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(17)) THEN
            nodg262=node_hw(index)
            gwdpth=(ells(nodg262)-h(nodg262))*s(nodg262)
```

```
            IF (pond(nodg262).ge.gwdpth) THEN
              stgg262=pond(nodg262)-gwdpth+ells(nodg262)
            ELSE
              stgg262=(pond(nodg262)-gwdpth)/s(nodg262)+ells(nodg262)
            END IF
C
            avvol=(stgg262-ells(nodg262)-deten(nodg262))*gdar/86400.
     &      *(1.0-carfact(nodg262)-total_small_afact(nodg262))
            avvol=amax1(avvol,0.0)
            IF (opt_l8_prop) THEN
              hdrop_adj=0.5
            ELSE
              hdrop_adj=0.0
            END IF
            stg_down=max(cstg_bkpt(index,1),
     &       chdep(idn_canal_index(index))+hdrop_adj)
            IF (stgg262.gt.stg_down) THEN
              capac=dischg_c(index)*(stgg262-stg_down)**expon(index)
            ELSE
              capac=0.0
            END IF
            cout=amin1(capac,avvol)
            dqu(idn_canal_index(index))=dqu(idn_canal_index(index))-
     &       cout
            dqu(idn_canal_index(index))=amax1(dqu(idn_canal_index(index)
     &       ),0.0)
            kflo(iflo)=cout
C-----------------------------------------------------
C                 culverts for L-8 restoration
C-----------------------------------------------------
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(18)) THEN
            IF (ihw_opt(index).eq.1) THEN
              nodl8r1=node_hw(index)
              gwdpth=(ells(nodl8r1)-h(nodl8r1))*s(nodl8r1)
              IF (pond(nodl8r1).ge.gwdpth) THEN
                stage=pond(nodl8r1)-gwdpth+ells(nodl8r1)
              ELSE
                stage=(pond(nodl8r1)-gwdpth)/s(nodl8r1)+ells(nodl8r1)
              END IF
              avvol=(stage-ells(nodl8r1)-deten(nodl8r1))*gdar/86400.
     &        *(1.0-carfact(nodl8r1)-total_small_afact(nodl8r1))
              avvol=amax1(avvol,0.0)
            END IF
            IF (opt_l8_prop) THEN
              hdrop_adj=0.50
            ELSE
              hdrop_adj=0.0
            END IF
            stg_down=max(cstg_bkpt(index,1),
     &       chdep(idn_canal_index(index))+hdrop_adj)
            head_diff=max(stage-stg_down,0.0)
            cout=dischg_c(index)*head_diff
            cout=amin1(cout,avvol)
            dqu(idn_canal_index(index))=dqu(idn_canal_index(index))-
     &       cout
            dqu(idn_canal_index(index))=amax1(dqu(idn_canal_index(index)
```

```
     &       ),0.0)
            kflo(iflo)=cout
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(19)) THEN
            IF (ihw_opt(index).eq.1) THEN
              nodl8r2=node_hw(index)
              gwdpth=(ells(nodl8r2)-h(nodl8r2))*s(nodl8r2)
              IF (pond(nodl8r2).ge.gwdpth) THEN
                stage=pond(nodl8r2)-gwdpth+ells(nodl8r2)
              ELSE
                stage=(pond(nodl8r2)-gwdpth)/s(nodl8r2)+ells(nodl8r2)
                avvol=(stage-ells(nodl8r2)-deten(nodl8r2))*gdar/86400.
     &          *(1.0-carfact(nodl8r2)-total_small_afact(nodl8r2))
                avvol=amax1(avvol,0.0)
              END IF
            ELSE
              stage=chdep(iup_canal_index(index))+
     &         hdc(iup_canal_index(index),iseas)
            END IF
            IF (opt_l8_prop) THEN
              hdrop_adj=0.50
            ELSE
              hdrop_adj=0.0
            END IF
            stg_down=max(cstg_bkpt(index,1),
     &       chdep(idn_canal_index(index))+hdrop_adj)
            head_diff=max(stage-stg_down,0.0)
            cout=dischg_c(index)*head_diff
            cout=amin1(avvol,cout)
            dqu(idn_canal_index(index))=dqu(idn_canal_index(index))-
     &       cout
            dqu(idn_canal_index(index))=amax1(dqu(idn_canal_index(index)
     &       ),0.0)
            kflo(iflo)=cout
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C---------------------------------------------
C               CULVERT 10A
C---------------------------------------------
C
          ELSE IF (flnm(iflo).eq.struc_name_spec_code(20)) THEN
C

            il8=idn_canal_index(index)
            total_l8_ag_dmnd = amax1(dmndagl8,0.0) + add_dmndagl8
            IF (.not.opt_l8_prop) THEN
              IF (stagelo.gt.chdep(il8)+hdc(il8,iseas)) THEN
                hdiff=stagelo-chdep(il8)-hdc(il8,iseas)
              ELSE
                hdiff=chdep(il8)+hdc(il8,iseas)-stagelo
              END IF
C
              IF (stagelo.gt.stg_bkpt(index,1)) THEN
                capl8=dischg_c(index)
              ELSE IF (stagelo.gt.stg_bkpt(index,2)) THEN
                capl8=dischg_c(index)/(stg_bkpt(index,1)-stg_bkpt(index,
```

```
     &           2))*(stagelo-stg_bkpt(index,2))
              ELSE
                capl8=0.0
              END IF
C
              capac=capacc10a_bak*sqrt(hdiff)
              IF (chdep(il8)+hdc(il8,iseas).gt.stagelo) THEN
                wsneedl8=dqu(il8)
                cout=0.0
              ELSE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                IF (stagelo.lt.sched(iday,month)) THEN
                  IF (total_l8_ag_dmnd.gt.0.0001) THEN
                    wsdel=amin1(dqu(il8),total_l8_ag_dmnd)*pctmet/1.9835
                    cout=amin1(wsdel,capac,capl8)
                    wsneedl8=dqu(il8)-cout
                  ELSE
                    cout=0.0
                    wsneedl8=dqu(il8)
                  END IF
                ELSE
                  wsdel=dqu(il8)
                  cout=amin1(wsdel,capac,capl8)
                  wsneedl8=dqu(il8)-cout
                END IF
              END IF
              IF (total_l8_ag_dmnd.gt.0.0001) THEN
                wsdml8=amin1(dqu(il8),total_l8_ag_dmnd/1.9835)
                wsdmndl8_not_met=amax1(wsdml8-cout,0.0)
                qu(il8)=qu(il8)-(total_l8_ag_dmnd/1.9835
     &           -wsdmndl8_not_met)*86400.
              ELSE
                wsdmndl8_not_met=0.0
              END IF
            ELSE
              cout=0.0
            END IF
c          PRINT *,COUT,DMNDAGL8/1.9835,IAGFLO(IAG(IL8AGBSN),1),STAGELO
c    &       ,CHDEP(IL8),WSDML8,WSDEL,CAPAC,CAPL8,DQU(IL8),'  IN ROUTE'
c    &       ,IL8,WSNEEDL8,IFLO,add_dmndagl8/1.9835,total_l8_ag_dmnd
            kflo(iflo)=cout
C
          END IF
        END IF
   50 CONTINUE
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      RETURN
      END
```

taycrnbs_res.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: taycrnbs_res.F,v 1.4 2003/07/03 20:07:49 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/taycrnbs_res.F,v $
C   $Name: rel-5-5 $
C
C
C  description:
C   SUBROUTINE to do water budget for proposed Taylor Creek/Nubbin Slough Res
C   preliminary specs are:
C     5kac @ 10ft max depth
C    Initial inflow pump capacity = 2500cfs (to be resized as necessary)
C    Location TBD, but not necessary to specifically define for SFWMM simulation
C    operations:
C     * Sources of inflow are outflows from S191 and S133 (Taylor Creek
C            and Nubbin Slough)
C     * Releases  to lake when lake stage is  > 0.4ft below the
C       bottom pulse zone.(max capacity is 1000 cfs)
C   note: Above operating conditions have to be defined in main program.
C
C  input parameters:
C   ares      = area or reservoir [acres]
C   hres_min  = minimum stage in reservoir [ft]
C   hres_max  = maximum stage in reservoir [ft]
C   hres0     = beginning of day stage [ft]
C   res_rf_day = reservoir rainfall depth [ft]
C   res_et_day = reservoir evapotranspiration IF water is available [ft]
C   qin_max   = reservoir inflow pump capacity [cfs]
C   qout_max  = reservoir outflow pump capacity [cfs]
C
C  output parameters:
C   actual_et = actual reservoir evapotranspiration [ft]
C   actual_seepage = actual reservoir seepage [ft]
C   res2lok   = release from reservoir to Lake Okeechobee [cfs]
C   infltores   = inflow to reservoir [cfs]
C   hres      = END of day reservoir stage [ft]
```

```
C
C  other variables:
C   hres1     = reservoir stage after rf,et&seepage but before spill [ft]
C   hres2     = reservoir stage after spill [ft]
C   tot_res_loss = sum of actual_et and actual_seepage [acre-ft]
C
C***  ALL CALCULATIONS DONE IN UNITS OF CFS-DAY, UON !!!
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE taycrnbs_res (time_series_flow ,ares, hres_min,
     & hres_max, hres0 ,res_rf_day, res_et_day, qin_max, qout_max,
     & actual_et, actual_seepage, res2lok, infltores, hres, stglok,
     & stgref_nstor_to_lok, month, ifyr, iyear, iday)

      REAL actual_et, actual_seepage, ares, hres, hres_max, hres_min,
     & hres0, infltores, qin_max, qout_max, res_et_day, res_rf_day,
     & res2lok

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  assume reservoir properties
C  let: constant reservoir seepage loss rate = .0114 ft per day
C  Same seepage rates for proposed St. Lucie and Caloosahatchee reservoirs.

      res_seepage = .0114

C initialize reservoir variables

      infltores = 0.0
      res2lok = 0.0
      actual_et = 0.0
      actual_seepage = 0.0
      spill_over = 0.0

c update water level in reservoir based on RF-ET-Seepage
c assumption: natural processes (RF, ET & Seepage) occur
c             before any lake water enters the reservoir
c             or any reservoir water is pumpes into lake
      hres = hres0 + res_rf_day - res_et_day - res_seepage
      actual_et = res_et_day
      actual_seepage = res_seepage
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
      IF (hres .LT. 0.) THEN
c calculate ET and Seepage to bring hres to zero
c assumption: ET occurs before Seepage
         h0_plus_rf = hres0 + res_rf_day
         IF (h0_plus_rf .GT. res_et_day) THEN
            actual_et = res_et_day
            actual_seepage = h0_plus_rf - actual_et
         ELSE
            actual_et = h0_plus_rf
            actual_seepage = 0.0
         ENDIF
```

```
         hres = 0.0
      ENDIF
      hres1 = hres
c
      IF (hres .GT. hres_max) THEN
         spill_over = (hres - hres_max) * ares
         hres = hres_max
c
c note: spill_over is assumed to be not subject to any structure
c       capacity constraints and is considered a loss term in
c       the reservoir water budget
      ENDIF
      hres2 = hres
      tot_res_loss = (actual_et+actual_seepage)*ares
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c fill reservoir with time series flows
      avail_res_sto = amax1(0.,hres_max-hres) * ares / 1.9835
      infltores = amin1(time_series_flow / 1.9835,qin_max,avail_res_sto)
      hres = hres + infltores * 1.9835 / ares
      time_series_flow = time_series_flow - infltores * 1.9835
c
c make reservoir release into lake
      IF (stglok .LT. stgref_nstor_to_lok) THEN
         avail_res_rel = amax1(0.,hres-hres_min) * ares / 1.9835
c.....RES2LOK:
         res2lok = amin1(qout_max,avail_res_rel)
         hres = hres - res2lok * 1.9835 / ares
      ENDIF
c
c output for unit 126 in acre-ft
      WRITE(126,'(i5,2i3,1x,15f10.3)') ifyr+iyear-1,month,iday ,
     $res_rf_day*ares,spill_over ,res_et_day*ares,actual_et*ares ,
     $res_seepage*ares,actual_seepage*ares ,tot_res_loss,res2lok*1.9835,
     $infltores*1.9835,hres0,hres1,hres2,hres
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2


      RETURN
      END
```

trg.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: trg.F,v 1.11 2005/06/10 13:29:22 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/trg.F,v $
C   $Name: rel-5-5 $
C
C
C  Trigger package for SFWMM.  started 1/5/93 by Dean Randall, WRMI.
C  additional modifications by C. Neidrauer & L. Brion LECPD, PD, SFWMD
C
C  The trigger package allows the user to to set minimum limits on heads
C  at selected nodes.  If these limits are exceeded during
C  a run, the pumpage at certain wells is cutback according to an input
C  cutback schedule.  The routine is completely data-driven, except that
C  zones must be rectangular.
C
C  The package works as follows:  The variable USE_TRIGGER is set to a
C  positive INTEGER value in the 1st line of the LECDEF file. (This
C  value is the unit number of the trigger module input file.)
C  A positive value tells the model to call the triggering routine.
C
C  The module works according to zones, which are user-specified rectangles.
C  In each zone there are trigger cells. If the the average head in a
C  trigger cell is lower than a standard for a prescribed period of time,
C  the program looks for each well in the zone and
C  reduces the pumpage for the next stress period by a fraction specified in
C  the input data.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE trigger_input (unit_trig, nch, min_lok_ssm_cnt)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'trig_lec_et_param.inc'
```

```
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'

      CHARACTER*7 check_default, t_type
      CHARACTER*200 filename

      INTEGER echo_trig, i, iout, nch, phase, unit_trig, trig, type,
     & zone, zone1

      REAL*4 def_cutback(max_n_type,max_n_phase)

      DATA iout/6/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  Start by READing the data file.  Skip the top three line of the file
C  so they may be used as comments.

      READ(unit_trig,*)
      READ(unit_trig,*)
      READ(unit_trig,*)
      READ(unit_trig,*) unit_trig_out,echo_trig, min_lok_ssm_cnt

      FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/trigecho'
      OPEN(echo_trig, file=FILENAME, status='unknown')
      WRITE(echo_trig,90)
      WRITE(echo_trig,130) unit_trig_out

      READ(unit_trig,*)
      READ(unit_trig,*)
      READ(unit_trig,*)

      WRITE(echo_trig,110) (i,i=1,max_n_phase)

c Note that num_use_type refers to the number of irrigation use types.
c The total number of use types is num_use_type + 1, where the FIRST
c use type in the trginput.dat file must ALWAYS BE public water supply.
c Also note that the values in cutback(zone,trig,type,phase) are cutback
c fractions for public water supply, BUT they are maximum net irrigation
c application rates (inches/month) for the other use types.
      DO type = 1,num_use_type + 1
         READ(unit_trig,*) (def_cutback(type,phase), phase=1,
     $   max_n_phase)
         WRITE(echo_trig,120) type,(def_cutback(type,phase), phase=1,
     $   max_n_phase)
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c READ and error check the number of zones
c
      READ(unit_trig,*)
      READ(unit_trig,*) n_zone
      WRITE(echo_trig,140) n_zone
      READ(unit_trig,*)
      IF (n_zone .GT. max_n_zone) THEN
```

```
         WRITE(iout,10) max_n_zone
         STOP
      ENDIF
c
      DO zone = 1, n_zone
         READ(unit_trig,*)
         READ(unit_trig,*) zone1
         WRITE(echo_trig,150) zone1
         IF (zone .NE. zone1) THEN
            WRITE(iout,50) zone, zone1
            STOP
         ENDIF
c
         READ(unit_trig,*) zone_corner(zone,1,1),zone_corner(zone,1,2),
     $   zone_corner(zone,2,1),zone_corner(zone,2,2)
         WRITE(echo_trig,160) zone_corner(zone,1,1), zone_corner
     $   (zone,1,2),zone_corner(zone,2,1), zone_corner(zone,2,2)
c
c Error check to make sure that the rectangle is within the grid.
c
         IF (zone_corner(zone,1,1) .GT. max_column .OR. zone_corner
     $   (zone,2,1) .GT. max_column .OR. zone_corner(zone,1,2) .GT.
     $   max_row .OR. zone_corner(zone,2,2) .GT. max_row) THEN
c
            WRITE(iout,20) zone
            STOP
         ENDIF
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c Read and error check the number of trigger cells in a zone.
c
         READ(unit_trig,*) n_trigger(zone)
         WRITE(echo_trig,170) n_trigger(zone)
         IF (n_trigger(zone) .GT. max_n_trig) THEN
            WRITE(iout,30) max_n_trig, zone
            STOP
         ENDIF
c
c First READ the trigger type (temp. variable is t_type) and check to see if
c we're using a GW head or a canal depth.  If it's GW head, we need to READ
c in the trigger cell location.  The first part of the if-block works for
c GW heads
c
         DO trig = 1, n_trigger(zone)
            READ(unit_trig,'(a5)') t_type
            IF (t_type .EQ. 'gwhd' .OR. t_type .EQ. 'GWHD') THEN
               trigger_type(zone,trig) = 0
               READ(unit_trig,*) tcell_col(zone,trig), tcell_row
     $         (zone,trig), (trig_value(zone,trig,phase),phase=1,
     $         max_n_phase)
               READ(unit_trig,*) (trig_period(zone,trig,phase), phase=1,
     $         max_n_phase)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c Check to see IF cell is in model grid.
c
               IF ( tcell_row(zone,trig) .GT. max_row .OR. tcell_col
```

```
     $         (zone,trig) .GT. max_column) THEN
                  WRITE(iout,40) trig, zone
                  STOP
               ENDIF
            ELSE
c
c If we get here, we're using canal depths to DO the triggering.
c Compare the canal name in T_TYPE to the list of canal names that have been
c READ in to find a match.  This gives the canal index to use for the
c trigger.  If there is no match, WRITE an error and STOP. Start by setting
c TRIGGER_TYPE to -1.  When we find a match, set TRIGGER_TYPE to the canal
c index, indicating that a match has been found, and that the trigger type
c is a canal depth.
c
               trigger_type(zone,trig) = -1
               DO i = 1, nch
                  IF (t_type .EQ. cnm(i)) trigger_type(zone,trig) = i
               END DO
               IF (trigger_type(zone,trig) .EQ. -1) THEN
                  WRITE(*,220) t_type
                  STOP
               ENDIF
               READ(unit_trig,*) (trig_value(zone,trig,phase),phase=1,
     $         max_n_phase)
               READ(unit_trig,*) (trig_period(zone,trig,phase), phase=1,
     $         max_n_phase)
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

c
c set 0.03 <= trig_period <= 1.0. (.03 = 1 day) Set minimum to avoid divide-by-z
c
            DO phase = 1, max_n_phase
               trig_period(zone,trig,phase) = amax1(.03, amin1(1.,
     $         trig_period(zone,trig,phase)))
            END DO
c
            WRITE(echo_trig,180) tcell_col(zone,trig), tcell_row
     $      (zone,trig), (trig_value(zone,trig,phase),phase=1,
     $      max_n_phase)
            WRITE(echo_trig,210) (trig_period(zone,trig,phase), phase=1,
     $      max_n_phase)
c
c Now READ the cutback factors for the zone. If the CHARACTER string
c default of DEFAULT appears in column 1-7, use the default values.
c If any other CHARACTER string appears in those columns, READ new
c values for that zone.
c
            READ(unit_trig,'(a7)') check_default
            WRITE(echo_trig,190) check_default
            IF (check_default .EQ. 'default' .OR. check_default .EQ.
     $      'DEFAULT') THEN
               DO type = 1, num_use_type + 1
                  DO phase = 1, max_n_phase
                     cutback(zone,trig,type,phase)= def_cutback
     $               (type,phase)
                  END DO
```

```
                  cutback_lok(type) = def_cutback(type,1)
               END DO
            ELSE
               READ(unit_trig,*)
               READ(unit_trig,*)
               DO type = 1, num_use_type + 1
                  READ(unit_trig,*) (cutback(zone,trig,type,phase),
     $            phase = 1,max_n_phase)
c
                  cutback_lok(type) = cutback(zone,trig,type,1)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c note: final value for cutback_lok(type) is cutback amount corresponding
c       to the very last zone&trigger specified to have non-default
c       values in the input file
                  WRITE(echo_trig,200) (cutback(zone,trig,type,phase),
     $            phase = 1,max_n_phase)
c
               END DO             !END DO for type loop
            ENDIF                 !ENDIF for default check
         END DO                   !END DO for READing trigger cells
c initialize cutback phase
         cutb_phase(zone) = 0
      END DO                      !END DO for READing data for each zone
c
c Now we're done READing the trigger data file. Close the file.
c
      close(unit_trig)
c
c Now OPEN trigger output file and WRITE header
c
      FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/trigoutp'
      OPEN(unit_trig_out, file=FILENAME, status='unknown')
clb      OPEN(unit_trig_out, file='trigoutp', status='unknown')
      WRITE(unit_trig_out,70)
c
      close(echo_trig)
      FILENAME = DATADIR(1:LENGTH(DATADIR)) // '/trigwell'
      OPEN(unit_trig_out+1, file=FILENAME, status='unknown')
clb      OPEN(unit_trig_out+1,file='trigwell',status='unknown')
      WRITE(unit_trig_out+1,80)
      RETURN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c FORMATs
c
   10 FORMAT(/ ' Trigger module FATAL ERROR 1'/
     $' number of zones exceeds dimension of',i3, /
     $' Either reduce zones or increase
     $ parameter MAX_N_ZONE and recompile')
   20 FORMAT(/ ' Trigger module FATAL ERROR 2'/
     $' zone corner outside model grid for zone', i4)
   30 FORMAT(/' Trigger module FATAL ERROR 3'/
     $' number of trigger cells exceeds dimension of',i3, ' in zone',i3/
     $' reduce number of triggers or increase',
     $' dimension and recompile')
   40 FORMAT(/' Trigger module FATAL ERROR 4 '/ ' trigger cell',i3,
     $' in zone',i3,' is outside grid')
```

```
   50 FORMAT(/' Trigger module FATAL ERROR 5 '/
     $' zone indices DO not match', 2i4)
   60 FORMAT(/' Trigger module FATAL ERROR 6 '/ ' trigger cell',i3,
     $' not in zone',i3)
   70 FORMAT(//' Trigger module output file'//
     $'              number              canal or trigger  zone',
     $' simulated  trigger'/,
     $' year period    days zone trigger col  row   phase phase',
     $'     level    level')
   80 FORMAT(//' Trigger module well pumping adjustment output file'//
     $'                  canal or                           ',
     $'   input     new'/
     $' year period well col  row  trigger zone phase cutback',
     $'   wellq   wellq')
   90 FORMAT(1x/'ECHO PRINT  of Trigger Module Input Data')
  100 FORMAT(i3,' head/drawdown switch')
  110 FORMAT('  default cutback table'/
     $'                        trigger_phase'/ 4x,'use_type',7i8)
  120 FORMAT(4x,i8,7f8.3)
  130 FORMAT(i4,' unit number for trigger output')
  140 FORMAT(/,i4,' number of zones in input data'//' zone data')
  150 FORMAT(i4,' zone number')
  160 FORMAT(6x,2(i4,',',i4,3x))
  170 FORMAT(i4,' number of triggers in zone')
  180 FORMAT(6x,i4,',',i4,7f8.3)
  190 FORMAT(1x,a7,/' actual cutback values')
  200 FORMAT(f10.3,6f8.3)
  210 FORMAT(15x,7f8.3)
  220 FORMAT(' No match found for canal  ',a6, ' in trigger file')
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
*****************************************************************************
c  Subroutine SET_CUTBACK computes which zones require pumping cutbacks.
c  Zones requiring cutbacks set a CUTB_PHASE switch  > 0.
c  If ANY zone requires cutbacks, lec_restrictions is set to TRUE.
c
*****************************************************************************
c
      SUBROUTINE set_cutback(iyr, period, days, cnm, lok_ssm_cnt ,
     $min_lok_ssm_cnt)
cbw 01-02-01      SUBROUTINE set_cutback(iyr, period, days, cnm, lok_ssm_cnt)
c
      include 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
c
      INTEGER days, iyr, period, phase, tcol, trig, trow, zone ,
     $cutb_code_trig
      CHARACTER*5 cnm(max_ncnls)
c
c Cycle through zones, THEN trigger cells
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      lec_restrictions = .FALSE.
```

```
      DO zone = 1,n_zone
         cutb_code(zone) = 0
c Initialize cutback phase per zone.  Note that IF a zone was in a cutback
c phase the previous month AND IF we're in the dry season months (Nov-May),
c THEN the current zone must stay in AT LEAST phase 1 this month.
c Otherwise, initial the cutback phase to zero (no restrictions) for the zone.
         IF ( cutb_phase(zone) .GT. 0 .AND. (period .GE. 11 .OR. period
     $   .LE. 4) ) THEN
            cutb_phase(zone) = 1
            cutb_code(zone) = cutb_code(zone) + 1
         ELSE
            cutb_phase(zone) = 0
         ENDIF
c
         IF ( (period .GE. 10 .OR. period .EQ. 1) .AND. lok_ssm_cnt .GE.
     $   min_lok_ssm_cnt ) THEN
cbw 01-02-01     &        lok_ssm_cnt.GE.7 ) THEN
            cutb_phase(zone) = 1
            cutb_code(zone) = cutb_code(zone) + 100
         ENDIF
c
         cutb_code_trig = 0
         DO trig = 1, n_trigger(zone)
c
c Loop by phases, backward, to find the most severe violation.
c If the head is less than the trigger value for a
c given phase we have a violation for that phase.  Store that phase in array
c CUTB_PHASE.  The first time that the program finds a violation is the most
c severe.  So kick out of the loop once it finds a violation.
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            DO phase = max_n_phase, 1, -1
               n_days_trig_period = days - nint((1. - trig_period(zone,
     $         trig,phase)) * float(days))

               ave_head = sum_head(zone,trig,phase) / float
     $         (n_days_trig_period)
               IF (ave_head .LE. trig_value(zone,trig,phase)) THEN
                  cutb_code_trig = 1
                  cutb_phase(zone) = max0(cutb_phase(zone),phase)

                  IF (cutb_phase(zone) .EQ. phase) keep_trig(zone) =
     $            trig
c
                  IF (trigger_type(zone,trig) .EQ.  0) THEN
                     tcol = tcell_col(zone,trig)
                     trow = tcell_row(zone,trig)
                     WRITE(unit_trig_out,20) iyr, period,
     $               n_days_trig_period, zone,trig,tcol, trow, phase,
     $               cutb_phase(zone), ave_head, trig_value(zone,trig,
     $               phase)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  ELSE
                     WRITE(unit_trig_out,30) iyr, period,
     $               n_days_trig_period, zone,trig,cnm(trigger_type
     $               (zone,trig)), phase, cutb_phase(zone), ave_head,
```

```
     $               trig_value(zone,trig,phase)
                  ENDIF
c
                  go to 10        !go to next trigger
               ENDIF              !ENDIF for head less than trigger
            END DO                !END DO for PHASE
   10    END DO                   !END DO for TRIG
clb>>
         IF ( cutb_code_trig .EQ. 1) cutb_code(zone) = cutb_code(zone)
     $   + 10
c
         IF ( cutb_code(zone) .EQ. 100 .OR. cutb_code(zone) .EQ. 101)
     $   THEN
c only lok ssm (or lok ssm+dry season) will trigger cutback next month
            WRITE(unit_trig_out,20) iyr, period, lok_ssm_cnt, zone, 0,
     $      0, 0, 1, 1

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c     &             ,0.0, 0.0
         ENDIF
         IF ( cutb_code(zone).EQ. 1) THEN
c only dry season will trigger cutback next month
            WRITE(unit_trig_out,20) iyr, period, lok_ssm_cnt, zone, -1,
     $      0, 0, 1, 1
c     &             ,0.0, 0.0
         ENDIF
clb<<
         IF (cutb_phase(zone) .GT. 0) lec_restrictions = .TRUE.
c
      END DO                        !END DO for ZONE
c
      RETURN
c
c
   20 FORMAT(i5,i7,i8,i5,i8,i4,i5,i8,i6,f10.2,f9.2)
   30 FORMAT(i5,i7,i8,i5,i8,4x,a5,i8,i6,f10.2,f9.2)
      END
c
c
c
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
*****************************************************************************
c  Subroutine REDUCE_WELLQ sets a value of FRACT_WELLQ, which is the fraction
c  of the input pumping that the reduced pumping must be.  This is done when
c  cutback switches are set (CUTB_PHASE > 0).  This SUBROUTINE is called for
c  each well, for each month.
c  For irrigation use types (type.GE.1), this SUBROUTINE RETURNs the maximum
c  NET irrigation withdrawal for the cell: nirrmax(location,use_type).
c  (units = feet over the cell)
*****************************************************************************
c
      SUBROUTINE reduce_wellq(iyr, period, location, type, well)
c
      include 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
```

```
      INCLUDE 'trigger.inc'
c
      INTEGER i, iyr, keep_zone, location
      INTEGER period, trig, type, x_loc, y_loc, zone, well
c
c First find the (x_loc, y_loc) location of the well from the LOCW value
c
      DO i = 1, maxy
         IF (location .LE. isum(i+1)) THEN
            y_loc = i
            x_loc = location + minx(i) - 1 - isum(i)
            GO TO 10
         ENDIF
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c We can have the situation were a well is in more than 1 zone,
c and cutbacks are set for each zone.  These cutbacks may be different values.
c One of the functions of the following piece of code is to find the maximum
c cutback among the zones that a well is a part of. note: still needs testing
c                                                        for this scenario
c Now cycle through all the zones. If a well is a part of a zone (check that
c the well coordinates are within the zone boundaries), and the
c well TYPE is > 0, and the CUTB_PHASE is > 0 (value of 0 indicates no
c cutbacks are necessary) THEN set a new value of FRACT_WELLQ.
c FRACT_WELLQ stores that value giving the maximum cutback for a well.
c KEEP_ZONE stores for output the zone that is in violation.
c
   10 fract_wellq(location) = 1.
c min_maxappl_rate applies to any particular use type & is in units of inches
      min_maxappl_rate(location) = 100.
      IF (type .NE. 0) THEN
         nirrmax(location,type) = ( min_maxappl_rate(location) / 12.) *
     $   frac_irr_area(location,type)
      ENDIF
c
      keep_zone = 0
      DO zone = 1,n_zone
         IF ( x_loc .GE. zone_corner(zone,1,1) .AND. x_loc .LE.
     $   zone_corner(zone,2,1) .AND. y_loc .GE. zone_corner(zone,1,2)
     $   .AND. y_loc .LE. zone_corner(zone,2,2) .AND. cutb_phase(zone)
     $   .GE. 1) THEN
clb>>
            IF ( cutb_code(zone).EQ. 100 .OR. cutb_code(zone) .EQ. 101)
     $      THEN
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c only LOK ssm (or lok ssm+dry season) was responsible for triggering cutbacks
c for this zone for this period; after the fract_wellq & nirrmax() are set,
c we need not look at the other zones by which this well is a member of
c therefore, exit this SUBROUTINE
               IF ( type.EQ. 0) THEN
                  fract_wellq(location) = 1.- cutback_lok(type+1)
c
                  WRITE(unit_trig_out + 1,40) iyr, period, well, x_loc,
     $            y_loc, 0, zone, 1, 1. - fract_wellq(location),
     $            pump_def(well,period), pump_def(well,period)
     $            * fract_wellq(location)
```

```
               ELSE
                  IF ( cutback_lok(type + 1) .LE. min_maxappl_rate
     $            (location) ) THEN
                     min_maxappl_rate(location) = cutback_lok(type + 1)

                  ENDIF
c nirrmax is min_maxappl_rate adjusted for fraction of use type actually in
c one cell
                  nirrmax(location,type) = ( min_maxappl_rate(location)
     $            /12. ) * frac_irr_area(location,type)

               ENDIF
               GO TO 20
            ENDIF       !END IF block IF this zone is triggered by lak
clb<<
clb>>
            IF ( cutb_code(zone) .EQ. 1) THEN
c only dry season was responsible for triggering cutbacks for this
c zone for this period; after the fract_wellq & nirrmax() are set,
c we need not look at the other zones by which this well is a member
c of therefore, exit this SUBROUTINE
               IF ( type .EQ. 0) THEN
                  fract_wellq(location) = 1. - cutback_lok(type + 1)
c
                  WRITE(unit_trig_out + 1,40) iyr, period, well, x_loc,
     $            y_loc, -1, zone, 1, 1. - fract_wellq(location),
     $            pump_def(well,period), pump_def(well,period)
     $            * fract_wellq(location)
               ELSE
                  IF ( cutback_lok(type + 1) .LE. min_maxappl_rate
     $            (location) ) THEN
                     min_maxappl_rate(location) = cutback_lok(type + 1)
                  ENDIF
c nirrmax is min_maxappl_rate adjusted for fraction of use type actually in
c one cell
                  nirrmax(location,type) = ( min_maxappl_rate(location)
     $            /12. ) * frac_irr_area(location,type)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               ENDIF
               GO TO 20
            ENDIF
c END IF block IF this zone is triggered by dry season (or dry season + lake
ssm)
clb<<
c case wherein at least a trigger well was responsible for imposing a cutback
c on this x_loc,y_loc ( cut_code(zone) = 010 or 011 or 110 or 111 )
            DO trig = 1, n_trigger(zone)
c Select largest cutback fraction for public water supply
c Note: type=0 is public water supply
               IF (type .EQ. 0) THEN
                  IF (1. - cutback(zone,trig,type + 1,cutb_phase(zone))
     $             .LE. fract_wellq(location)) THEN
                     fract_wellq(location) = 1.- cutback(zone,trig,type
     $               + 1,cutb_phase(zone))
                     keep_zone = zone
                  ENDIF
               ELSE
```

```
c The variable, cutback(zone,trig,type+1,cutb_phase(zone)), is a
c maximum NET monthly application rate for irrigation uses[inches/month].
c Select the smallest (ie, most restrictive) max application rate to compute
c the allowable maximum NET irrigation withdrawal for the cell [feet].
                  IF ( cutback(zone,trig,type + 1,cutb_phase(zone)) .LE.
     $            min_maxappl_rate(location) ) THEN
                     min_maxappl_rate(location) = cutback(zone,trig,type
     $               + 1,cutb_phase(zone))
                  ENDIF
c nirrmax is min_maxappl_rate adjusted for fraction of use type actually in
c one cell
                  nirrmax(location,type) = ( min_maxappl_rate(location)
     $            / 12. ) * frac_irr_area(location,type)

               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            END DO     !END loop for triggers within zone
         ENDIF          !END IF block for check IF location is in zone
      END DO             !END loop for zone
c
c If KEEP_ZONE > 0, we have a cutback.
c Write out stuff IF there's a cutback for PWS (type=0)
c Don't DO WRITEs for irrigation use types.
c
      IF (keep_zone .GT. 0 .AND. type .EQ. 0) THEN
         WRITE(unit_trig_out + 1,40) iyr, period, well, x_loc, y_loc,
     $   keep_trig(keep_zone), keep_zone, cutb_phase(keep_zone), 1. -
     $   fract_wellq(location), pump_def(well,period), pump_def(well,
     $   period)*fract_wellq(location)
      ENDIF
c
   20 RETURN
   30 FORMAT(i5,i7,i5,i4,i5,i9,i5,i6,f8.3,2i8)
   40 FORMAT(i5,i7,i5,i4,i5,i9,i5,i6,f8.3,2f8.3)
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c ***************************************************************************
c  This SUBROUTINE sums heads in trigger cells for a specified number of
c   days during the month
c ***************************************************************************
c
      SUBROUTINE sum_trig_heads(days)
c
      include 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'trigger.inc'
c
      INTEGER cell, days, phase, start_day, tcol, trig, trow, zone
c
      DO zone = 1, n_zone
         DO trig = 1, n_trigger(zone)
            IF (trigger_type(zone,trig) .EQ. 0) THEN
               tcol = tcell_col(zone,trig)
               trow = tcell_row(zone,trig)
```

```
               cell = tcol - minx(trow) + 1 + isum(trow)
               DO phase = 1, max_n_phase
                  start_day =  1 + nint((1. - trig_period(zone,trig,
     $            phase)) * float(days))
                  IF (iday .GE. start_day) THEN
                     sum_head(zone,trig,phase) = sum_head(zone,trig,
     $               phase) + ( h(cell) + pond(cell) )
                  ENDIF
               END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            ELSE
               DO phase = 1, max_n_phase
                  start_day =  1 + nint((1. - trig_period(zone,trig,
     $            phase)) * float(days))
                  IF (iday .GE. start_day) THEN
                     sum_head(zone,trig,phase) = sum_head(zone,trig,
     $               phase) + chdep(trigger_type(zone,trig))
                  ENDIF
               END DO
            ENDIF
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      RETURN
      END
```

trig_lec_et_param.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: trig_lec_et_param.inc,v 1.2 2003/07/03 20:08:07 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/trig_lec_et_param.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
c This file contains parameters used by the trigger and LEC ET routines
c
      integer max_n_zone, max_n_trig, max_n_type, max_n_phase
      parameter (max_n_zone=32, max_n_trig=50, max_n_type=10,
     *           max_n_phase=4)
```

trigger.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: trigger.inc,v 1.3 2003/07/03 20:08:08 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/trigger.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
c Trigger module common block declaratives, here through the COMMON statment
c

      integer max_column, max_row, n_zone, unit_trig_out
      integer use_trigger
      integer cutb_phase(max_n_zone),cutb_code(max_n_zone)
      integer keep_trig(max_n_zone), n_trigger(max_n_zone)
      integer tcell_col(max_n_zone,max_n_trig)
      integer tcell_row(max_n_zone,max_n_trig)
      integer trigger_type(max_n_zone, max_n_type)
      integer zone_corner(max_n_zone,2,2)
      logical lec_restrictions,lec_restrictions1

      real*4 cutback_lok(max_n_type)
      real*4 cutback(max_n_zone, max_n_trig, max_n_type, max_n_phase)
      real*4 sum_head(max_n_zone, max_n_trig, max_n_phase)
      real*4 trig_period(max_n_zone, max_n_trig, max_n_phase)
      real*4 trig_value(max_n_zone, max_n_trig, max_n_phase)
      real*4 fract_wellq(max_n_wells)

c
      common /trigger/ cutb_phase, keep_trig, max_column, max_row,
     *        n_trigger, n_zone, tcell_col, tcell_row,  unit_trig_out,
     *        use_trigger, zone_corner, cutback, sum_head,
     *        trig_period, trig_value, trigger_type, fract_wellq,
     *        lec_restrictions,cutback_lok,cutb_code,lec_restrictions1
c ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

val2val_w_lok.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: val2val_w_lok.F,v 1.9 2003/07/03 20:07:50 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/val2val_w_lok.F,v $
C   $Name: rel-5-5 $
C
C
C  this program creates an array containing active SFWMM cell
C  stage-lsel values plus lake_stage-bathymetry values
C  no. of SFWMM active cells = 1746
C  no. of LOK cells = 176
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE val2val_w_lok (iopt, stagelo, stagelores, isum_w_lok,
     & minx_w_lok, maxx_w_lok)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'addlok.inc'

      INTEGER isum_w_lok(max_n_rows_incl_lok),
     & maxx_w_lok(max_n_rows_incl_lok), maxxlok(max_n_rows_lok),
     & minx_w_lok(max_n_rows_incl_lok), minxlok(max_n_rows_lok),
     & minxlokres(max_n_rows_lok)

      REAL bathymetry(max_n_cells_lok)

      DATA bathymetry /10.59,11.49,10.94,12. 12,11.95, 9.53, 8.21,
     & 8.60, 9.09, 12.30, 12.40, 11.09, 9.14, 6.85, 5.50, 5.31, 5.20,
     & 8.48, 9.87, 12.90, 12.67, 8.90, 6.87, 6.10, 6.01, 4.93, 4.75,
     & 3.98, 6.54, 13.46, 12.93, 13.53, 12.04, 7.32, 6.81, 5.84, 3.08,
     & 2.33, 2.08, 1.51, 2.14, 14.14, 13.49, 12.84, 12.89, 11.20, 7.27,
     & 5.35, 2.51, 0.84, 0.29, 0.43, 0.67, 0.76, 1.66, 13.82, 13.59,
```

```
     & 12.91, 11.48, 11.96, 8.97, 4.99, 2.85, 1.43, 0.61, -0.19, -0.39,
     & -0.32, 0.19, 1.44, 3.22, 13.56, 12.98, 11.78, 11.09, 10.73, 5.39,
     & 3.22, 1.85, 0.53, -0.69, -0.45, -0.04, -0.17, 0.29, 0.95, 2.35,
     & 12.91, 11.53, 10.66, 9.98, 8.60, 4.49, 3.80, 2.25, 0.73, 0.33,
     & 0.66, 0.98, 1.03, 1.28, 1.63, 3.27, 11.92, 10.54, 9.57, 9.58,
     & 5.80, 3.83, 3.21, 1.78, 1.31, 1.32, 1.40, 1.39, 1.51, 1.72, 4.56,
     & 12.74, 12.13, 9.78, 4.83, 3.54, 3.13, 1.58, 1.47, 1.41, 1.45,
     & 1.82, 2.49, 6.29, 13.55, 12.81, 9.70, 5.75, 4.54, 3.02, 1.77,
     & 1.06, 1.34, 2.00, 3.80, 14.17, 13.31, 11.65, 8.12, 5.65, 3.53,
     & 1.83, 2.08, 3.23, 7.17, 14.08, 13.08, 12.02, 10.86, 4.90, 2.46,
     & 2.67, 5.71, 10.99, 12.80, 11.55, 8.50, 4.14, 2.36, 4.37, 9.56,
     & 10.57, 8.82, 4.44, 3.70, 7.37, 9.12, 6.31, 7.45/
      DATA maxxlok /19, 20, 20, 20, 20, 21, 22, 22, 22,  22, 22, 21, 21,
     & 21, 20, 19, 18/
      DATA minxlok /16, 14, 13, 11, 9, 8, 7, 7, 7,  8, 10, 11, 12, 13,
     & 14, 15, 16/
      DATA minxlokres /16, 14, 13, 11, 17, 16, 15, 14, 14,  15, 16, 17,
     & 18, 18, 18, 18, 18/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      maxylok = 71
      minylok = 55
      minylokres = 59

      IF (iopt .EQ. 0) THEN
c >>> build isum, minx, maxx arrays for lok and w_lok systems
c
c note: first lokrow is in irow = 55
c       i.e., irowlok = irow-minylok+1
c
c build isum array for lok
         isumlok(1) = 0
         DO i = 1,maxylok-minylok + 1
            isumlok(i+1) = isumlok(i) + maxxlok(i) - minxlok(i) + 1
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c build minx & maxx arrays for updated (w_lok) system
c
c from row 1 to minylok-1
         DO irow = 1,minylok-1
            minx_w_lok(irow) = minx(irow)
            maxx_w_lok(irow) = maxx(irow)
         END DO
c
c from row minylok to maxy
         DO irow = minylok,maxy
            irowlok = irow-minylok+1
            minx_w_lok(irow) = minxlok(irowlok)
            maxx_w_lok(irow) = maxx(irow)
         END DO
c
c from row maxy+1 to maxylok
         DO irow = maxy+1,maxylok
            irowlok = irow-minylok+1
            minx_w_lok(irow) = minxlok(irowlok)
```

```
            maxx_w_lok(irow) = maxxlok(irowlok)
         END DO
c
c build isum array for updated (w_lok) system
c
c from row 1 to minylok
         DO irow = 1,minylok
            isum_w_lok(irow) = isum(irow)
         END DO
c
c from row minylok+1 to maxy+1
         DO irow = minylok,maxy
            irowlok = irow-minylok+1
            isum_w_lok(irow+1) = isum_w_lok(irow) + maxx(irow) - minxlok
     $      (irowlok) + 1
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c from row maxy+2 to maxylok+1
         DO irow = maxy+1,maxylok
            irowlok = irow-minylok+1
            isum_w_lok(irow+1) = isum_w_lok(irow) + maxxlok(irowlok) -
     $      minxlok(irowlok) + 1
         END DO
c
      ELSE
c >>> create values_w_lok array
c
c from row 1 to minylok-1
c can DO direct assignment for all cells below the lake
c         nmax = isum_w_lok(minylok-1)
c     &        + maxx_w_lok(minylok-1) - minx_w_lok(minylok-1) + 1
c nmax == nodemax for this case
         nodemax = isum(minylok-1) + maxx(minylok-1) - minx(minylok-1)
     $   + 1
c         print *,'nmax-nodemax:',nmax-nodemax
         DO node = 1,nodemax
c node_w_lok == node for this case
            values_w_lok(node) = values(node)
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
c from row minylok to maxy
c have to DO row-by-row assignment
         DO irow = minylok,maxy
            irowlok = irow-minylok+1
c load data from lok_side
            DO icol = minxlok(irowlok),maxxlok(irowlok)
               node_w_lok = isum_w_lok(irow) + icol - minx_w_lok(irow)
     $             + 1
               nodelok = isumlok(irowlok) + icol - minxlok(irowlok) + 1
               values_w_lok(node_w_lok) = stagelo - bathymetry(nodelok)
            END DO
c load data from sa1_side
            DO icol = minx(irow),maxx(irow)
               node_w_lok = isum_w_lok(irow) + icol - minx_w_lok(irow)
     $         + 1
```

```
               node = isum(irow) + icol - minx(irow) + 1
               values_w_lok(node_w_lok) = values(node)
            END DO
         END DO
c
c from row maxy+1 to maxylok
c have to DO row-by-row assignment
         DO irow = maxy+1,maxylok
            irowlok = irow-minylok+1
c load data from lok_side only
            DO icol = minxlok(irowlok),maxxlok(irowlok)
               node_w_lok = isum_w_lok(irow) + icol - minx_w_lok(irow)
     $         + 1
               nodelok = isumlok(irowlok) + icol - minxlok(irowlok) + 1
               values_w_lok(node_w_lok) = stagelo - bathymetry(nodelok)
            END DO
         END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
clb w/ lok res>>
         IF (stagelores.NE.-999.) THEN
c Lake okeechobee is split into 2 compartments, modify depths
c for reservoir compartment (east side)  of Lake Okeechobee.
            DO irow = minylokres,maxylok
               irowlok = irow-minylok+1
               DO icol = minxlokres(irowlok),maxxlok(irowlok)
                  node_w_lok = isum_w_lok(irow) + icol - minx_w_lok
     $            (irow) + 1
                  nodelok = isumlok(irowlok) + icol - minxlok(irowlok)
     $            + 1
                  values_w_lok(node_w_lok) = stagelores - bathymetry
     $            (nodelok)
               END DO
            END DO
         ENDIF
clb w/ lok res<<
      ENDIF
c
      RETURN
      END
```

wca_avail_stor_dmnd_setup.F

```
C-----------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-----------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: wca_avail_stor_dmnd_setup.F,v 1.12 2005/06/10 13:29:22 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wca_avail_stor_dmnd_setup.F,v $
C   $Name: rel-5-5 $
C
C-----------------------------------------------------------------------------
C     This subroutine computes available storage in the Conservation
C     areas,which include Lake Okeechobee releases,for flood control
C     purposes and for meeting demands in Lower East Coast Service Areas
C     and  Everglades National Park.
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE wca_avail_stor_dmnd_setup (tinflow, add_need,
     & total_outf_frm_cnl, reg_rel_frm_lok, outflow_from_3a, tneed_orig,
     & wsneedl8, dqu, avail_vol_pipel, tneed_pipel, nsvarea, month,
     & avvolws, avvolrg, climate_index, avvol_rlok_to_sa1,
     & ws_from_lo_s5a, ws_from_lo_s6, regcoffset, lwslo_floor, chdep,
     & frf, ds_convey_c_name, wsnde1, wsnde2, wsnde3, wsndes, wsndese,
     & ibot_zone_regd, jday)

      INCLUDE  'wmm_par.inc'
      INCLUDE  'wcaot.inc'
      INCLUDE  'wcasim.inc'
      INCLUDE  'routc.inc'
      INCLUDE  'stat.inc'
      INCLUDE  'climvar.inc'
      INCLUDE  'opersched.inc'
      INCLUDE  'loktarg.inc'
      INCLUDE  'c1.inc'
      INCLUDE  'c2.inc'

      DIMENSION add_need(max_n_wcas,max_n_conv_cnl_wca),
     & avail_vol_pipel(max_n_eaa_condt),
     & avvolrg(max_n_wcas,max_n_conv_cnl_wca),
```

```
     & avvolws(max_n_wcas,max_n_conv_cnl_wca), chdep(max_ncnls),
     & climate_index(12), dqu(max_ncnls),
     & ibot_zone_regd(max_n_stor_w_oper_sch), lwslo_floor(max_n_wcas),
     & outflow_from_3a(max_n_conv_cnl_wca),
     & reg_rel_frm_lok(max_n_eaa_condt), regcoffset(max_n_wcas),
     & tinflow(max_n_wcas,max_n_conv_cnl_wca),
     & tneed_orig(max_n_wcas,max_n_conv_cnl_wca),
     & tneed_pipel(max_n_eaa_condt),
     & total_outf_frm_cnl(max_n_wcas,max_n_conv_cnl_wca)

      CHARACTER ds_convey_c_name(max_n_eaa_condt)*5

      LOGICAL lwslo_floor
c      LOGICAL lbotschedm

      DOUBLE PRECISION chdep,stgminwca
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C   INITIALIZE VARIABLES

      DO isa=1,nwcas
        DO i=1,n_intcnl(isa)
          tneed(isa,i)=0.0
          tinflow(isa,i)=0.0
          add_need(isa,i)=0.0
          total_outf_frm_cnl(isa,i)=0.0
        END DO
      END DO

      reg_rel_frm_lok(1)=regs354
      reg_rel_frm_lok(2)=nnrreg
      reg_rel_frm_lok(3)=regs352
      reg_rel_frm_lok(4)=hlsbreg
C
      IF (n_intcnl(iwca3a).gt.0) THEN
        DO i=1,n_intcnl(iwca3a)
          outflow_from_3a(i)=0.0
        END DO
      END IF
C
C  CALCULATE THE TOTAL LEC SURFACE WATER NEEDS TO BE MET BY
C  EACH OF WCAS. INFORMATION IS NEEDED FOR PRO-RATING
C  OF AVAILABLE SUPPLY IF SUPPLY IS NOT SUFFICIENT TO
C  MEET THE NEEDS
C
      IF (opt_l8_prop) THEN
        wsneed_l8=dqu(il8s)
      ELSE
        wsneed_l8=wsneedl8
      END IF
      IF (iconvey_canal_no_lwdd.ne.0) THEN
        add_need(idn_wca_name_lwdd_indx,iconvey_canal_no_lwdd)=
     &   add_need(idn_wca_name_lwdd_indx,iconvey_canal_no_lwdd)+wsnde1+
     &   wsnde2+wsnde3+wsndese
        add_need(idn_wca_name_l8_indx,iconvey_canal_no_l8)=
     &   add_need(idn_wca_name_l8_indx,iconvey_canal_no_l8)+wsneed_l8
      END IF
```

```
      add_need(idn_wca_indx(is150bws-6),iconvey_canal_no(is150bws-6))=
     &add_need(idn_wca_indx(is150bws-6),iconvey_canal_no(is150bws-6))+
     &frf
      DO isa=1,nwcas
        DO i=1,n_intcnl(isa)
          IF (nclns(isa,i).gt.0) THEN
            DO j=1,nclns(isa,i)
              tneed(isa,i)=tneed(isa,i)+dqu(isacnl(isa,i,j))
            END DO
            tneed(isa,i)=tneed(isa,i)+add_need(isa,i)
          END IF
          tneed_orig(isa,i)=tneed(isa,i)
        END DO
      END DO
C
C   DETERMINE TOTAL STRUCTURE INFLOW FOR EACH OF WCAS
C
      CALL match (ds_convey_c_name(is6bsn-6), 1, 5, cnm, nch+1, 5,
     &iconv_1)
      CALL match (ds_convey_c_name(is7bsn-6), 1, 5, cnm, nch+1, 5,
     &iconv_2)
      CALL match (ds_convey_c_name(is150bws-6), 1, 5, cnm, nch+1, 5,
     &iconv_3)
      addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))=
     &addflw(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))+
     &asr_rec_vol_wca(lcnb(iconv_1))
      addflw(idn_wca_indx(is7bsn-6),iconvey_canal_no(is7bsn-6))=
     &addflw(idn_wca_indx(is7bsn-6),iconvey_canal_no(is7bsn-6))+
     &asr_rec_vol_wca(lcnb(iconv_2))
      addflw(idn_wca_indx(is150bws-6),iconvey_canal_no(is150bws-6))=
     &addflw(idn_wca_indx(is150bws-6),iconvey_canal_no(is150bws-6))+
     &asr_rec_vol_wca(lcnb(iconv_3))
      DO isa=1,nwcas
        DO i=1,n_intcnl(isa)
          tinflow(isa,i)=tinflow(isa,i)+addflw(isa,i)
          IF (nstrinf(isa,i).gt.0) THEN
            DO j=1,nstrinf(isa,i)
              tinflow(isa,i)=tinflow(isa,i)+kflo(istrindx(isa,i,j))
            END DO
          END IF
        END DO
C
C  determine demands and available water for conceptual pipelines
C   simulated
C
        DO i=1,nsvarea
          tneed_pipel(i)=0.0
          DO j=1,nsacnl(i)
            tneed_pipel(i)=tneed_pipel(i)+dqu(isa_canal_indx(i,j))
          END DO
          IF (i.eq.1) tneed_pipel(i)=tneed_pipel(i)+wsnde1+wsnde2+
     &     wsnde3+wsndese+wsneed_l8
          IF (i.eq.3) tneed_pipel(i)=tneed_pipel(i)+frf
          IF (i.eq.1) avail_vol_pipel(i)=ws_pipeline_ca1+
     &     ws_pipeline_ca1_hill+ws_pipeline_ca1_wpb
          IF (i.eq.2) avail_vol_pipel(i)=ws_pipeline_2a
          IF (i.eq.3) avail_vol_pipel(i)=ws_pipeline_3a+
```

```
     &     ws_pipeline_3a_s150
        END DO
C
C  DETERMINE THE AVAILABLE SUPPLY OF WATER FOR EACH WCA
C
C
        IF (wcaenv(isa).eq.'FALSE') THEN
          IF ((month.ge.10.or.month.le.5).and.climate_index(month)
     &     .lt.1.5.and.nzone(isa+1).gt.0) THEN
            regstg_wca(isa,1)=regstg_wca(isa,nzone(isa+1)-2)
          END IF
          stgminwca=max(regstg_wca(isa,1)-regcoffset(isa),stgmnrg(isa))
c       if (isa.eq.1) print *,regstg_wca(isa,1),stgmnrg(isa),' CA1REG  '
c    &     ,stgminwca
        END IF
C
C   FOR REGULATORY RELEASES
C
        DO i=1,n_intcnl(isa)
          IF (wcaenv(isa).eq.'FALSE') THEN
            IF (intcnl(isa,i).le.nch) THEN
              cnlstrg=(chdep(intcnl(isa,i))-stgminwca)*cl(intcnl(isa,i))
     &         *width(intcnl(isa,i))
              avvolrg(isa,i)=cnlstrg+cgseep(intcnl(isa,i),stgminwca,0,
     &         0.)+max(tovflc(intcnl(isa,i),stgminwca,0,0.,0,1),0.0)
     &         +tinflow(isa,i)*86400.
              avvolrg(isa,i)=amax1(avvolrg(isa,i),0.0)
              avvolrg(isa,i)=avvolrg(isa,i)/86400.
            END IF
          ELSE
            avvolrg(isa,i)=99999999.
          END IF
C
C   FOR WATER SUPPLY
C
C
          IF (lwslo_floor(isa)) THEN
            avstor=0.0
            avvolws(isa,i)=tinflow(isa,i)
          ELSE
            IF (intcnl(isa,i).le.nch) THEN
              cnlstws=(chdep(intcnl(isa,i))-floor_elev(isa,i))*
     &         cl(intcnl(isa,i))*width(intcnl(isa,i))
              avstor=cnlstws+cgseep(intcnl(isa,i),floor_elev(isa,i),0,
     &         0.)+tovflc(intcnl(isa,i),floor_elev(isa,i),0,0.,0,1)
C
              avvolws(isa,i)=amax1(avstor,0.0)+tinflow(isa,i)*86400.
              avvolws(isa,i)=amax1(avvolws(isa,i),0.0)
              avvolws(isa,i)=avvolws(isa,i)/86400.
            ELSE
              avvolws(isa,i)=tinflow(isa,i)
            END IF
          END IF
        END DO
        IF (wca_name(isa).eq.'WCA1 '.and.iconvey_canal_no(is5absn-6)
     &   .ne.0) THEN
          avvol_rlok_to_sa1=avvolws(isa,iconvey_canal_no(is5absn-6))-
```

```
     &     ws_from_lo_s5a-ws_from_lo_s6
          avvol_rlok_to_sa1=amax1(avvol_rlok_to_sa1,0.0)
        END IF
      END DO
C
      iwca=idn_wca_indx_env(is8bsn-6)
C
c     IF (trig_wca_stg(iwca3a).ge.regstg_wca(iwca3a,
c    &itop_zone_indx(iwca3a+1))+0.5.or.lbotschedm) THEN
c       lbotschedm=.true.
c     END IF
c     lbotschedm=.not.(trig_wca_stg(iwca3a).lt.regstg_wca(iwca3a,
c    &itop_zone_indx(iwca3a+1)))
c     IF ((rmulti_seas_pred(month).gt.rmulti_seas_thres_chg_sched(iwca)
c    &.or.(no_of_hurr_pred.ge.no_pred_storms_thres_wca.and.(month.ge.8.a
c    &nd.month.le.10))).or.lbotschedm) THEN
      regd=regstg_wca(iwca,ibot_zone_regd(iwca+1))
c     ELSE
c       regd=regstg_wca(iwca,ibot_zone_indx(iwca+1))
c     END IF
C
      IF (trig_wca_stg(iwca).gt.regd) THEN
        reg_comp=(trig_wca_stg(iwca)-regd)*rfregcf(month)
        IF (type_flow_across_ttrail.eq.'NSMFLOW') THEN
          IF (type_target.eq.'TOTAL') THEN
            totf=amax1(frf,reg_comp)
          ELSE
            total_target=srsdmnd_e(jday)+srsdmnd_w(jday)
            ratio_e=srsdmnd_e(jday)/total_target
            totf=amax1(total_target,reg_comp)
            totf_e=ratio_e*totf
            totf_w=(1.0-ratio_e)*totf
          END IF
        ELSE
          totf=frf+reg_comp
          IF (type_target.ne.'TOTAL') THEN
            totf_w=frf*rfcfs12(iseas)+reg_comp*rfcfs12_reg(iseas)
            totf_e=frf*rfcfs355(iseas)+reg_comp*rfcfs355_reg(iseas)
            srsdmnd_e(jday)=frf*rfcfs355(iseas)
            srsdmnd_w(jday)=frf*rfcfs12(iseas)
          END IF
        END IF
      ELSE
        reg_comp=0.0
        IF (type_flow_across_ttrail.eq.'NSMFLOW') THEN
          IF (type_target.eq.'TOTAL') THEN
            totf=frf
          ELSE
            totf_e=srsdmnd_e(jday)
            totf_w=srsdmnd_w(jday)
            totf=totf_w+totf_e
          END IF
        ELSE
          IF (type_target.eq.'TOTAL') THEN
            totf=frf
          ELSE
            totf_w=frf*rfcfs12(iseas)
```

```
            totf_e=frf*rfcfs355(iseas)
            totf=totf_w+totf_e
            srsdmnd_e(jday)=totf_e
            srsdmnd_w(jday)=totf_w
          END IF
        END IF
      END IF
C
      RETURN
      END
```

wcaot.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: wcaot.inc,v 1.5 2003/07/03 20:08:08 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wcaot.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      common /WCAOT/ cstg_bkpt(max_n_str_node_dep,5),
     &offset(max_n_targ_area,2),BFLWCF,BFLCNST,
     &RMXSTGE(max_n_ostruc_wca,max_n_wcas,5),
     &CAPFACT(max_n_ostruc_wca,max_n_wcas),
     &DCOEFF(max_n_ostruc_wca,max_n_wcas),
     &DESGNCP(max_n_ostruc_wca,max_n_wcas),
     &ICNLHW(max_n_ostruc_wca,max_n_wcas),
     &ICNLTW(max_n_ostruc_wca,max_n_wcas),
     &IHWOPT(max_n_ostruc_wca,max_n_wcas),
     &INTCNL(max_n_ostruc_wca,max_n_wcas),
     &ISACNL(max_n_ostruc_wca,max_n_wcas,max_n_str_inf_wca),
     &ISTRCPT(max_n_ostruc_wca,max_n_wcas),
     &ISTRINDX(max_n_wcas,max_n_conv_cnl_wca,max_n_str_inf_wca),
     &ITWOPT(max_n_ostruc_wca,max_n_wcas),
     &IXHW(max_n_ostruc_wca,max_n_wcas),
     &IXTW(max_n_ostruc_wca,max_n_wcas),
     &IYHW(max_n_ostruc_wca,max_n_wcas),
     &IYTW(max_n_ostruc_wca,max_n_wcas),
     &NBPT(max_n_ostruc_wca,max_n_wcas),
     &NCLNS(max_n_wcas,max_n_conv_cnl_wca),
     &NSTRINF(max_n_wcas,max_n_conv_cnl_wca),NWCAS,
     &POWER(max_n_ostruc_wca,max_n_wcas),RFCFS12(max_n_seas),
     &RFCFS355(max_n_seas),RFREGCF(12),
     &STGMNRG(max_n_wcas),IFLS39,
     &IFLS38,IFLS143,IFLS11,reg3a,
     &stg_bkpt(max_n_str_node_dep,5),IFLS144,IFLS145,IFLS146,IFLS141,
     &IFLS142,IFLS34,addflw(max_n_wcas,max_n_conv_cnl_wca),IFLS151,
     &IFLS31,IFLS333,IFLS334,IFLS339,IFLS340,IFLS12,ifls12a,IFLS346,
     &IFLS345,IFLS355,IG3A2,TNEED(max_n_wcas,max_n_conv_cnl_wca),
```

```
&ifls12b,IG3A9,IG3B2,IG3B3,IG3B4,
&IFLPSAC,NSTRPT(max_n_wcas),IPRORTE(max_n_wcas),IC11,IC304,
&IPWCA(max_n_wcas),IC11W,IL38E,TFLG123,RATFLW,TFLS9,
&offset_out(max_n_targ_area,2),DRAINC11W,LS9PUMP,LS11FLOW,REGS39,
&WSS39,REGS38,WSS38,REGS34,
&WSS34,REGS151,WSS151,WSS31,srsdmnd_w(366),ifls12c,srsdmnd_e(366),
&inesrs,iwenp,s12_env,s333_env,s355_env,ilv33,s12_reg,s333_reg,
&s355_reg,TOTF_E,TOTF_W,rega,regc,regd,ibcnp,
&n_trigger_fc_cnl(max_ncnls,max_nostr_cnl),
&icnl_no_fc(max_ncnls,max_nostr_cnl,3),
&n_trig_for_oper(max_ncnls,max_nostr_cnl,3),
&stg_trig_oper(max_ncnls,max_nostr_cnl,3,5),ilv37,
&outflow_from_roten_to_holy,idn_cnl_ws(max_ncnls,max_nostr_cnl),
&istr_s151,ifls12d,no_ds_cnl_ws(max_ncnls,max_nostr_cnl),
&idn_ws_cnl(max_n_ostruc_wca,max_n_wcas,5),
&no_dsws_cnl(max_n_ostruc_wca,max_n_wcas),
&total_frac_s12(5,2),
&RFCFS355_reg(max_n_seas),RFCFS12_reg(max_n_seas),
&iwca_reg_zone(max_n_ostruc_wca,max_n_wcas),
&total_frac_s12_reg(5,2),REGS31,outf_to_c9res_via_us27s,
&igrid_loc_constrnt
&(max_n_ostruc_wca,max_n_wcas,max_n_cells_constrt),
&iopt_for_tw_constraint(max_n_ostruc_wca,max_n_wcas),
&asr_rec_vol_wca(max_n_wcas),
&iopt_for_lokreg_flwth(max_n_ostruc_wca,max_n_wcas),
&frac_ws_bkp_abv_wcaflr(max_n_eaa_condt),
&frac_ws_bkp_bel_wcaflr(max_n_eaa_condt),
&sim_opt(max_n_ostruc_wca,max_n_wcas),
&icode(max_n_str_node_dep)
  LOGICAL LS9PUMP,LS11FLOW
  CHARACTER icode*3,sim_opt*3
  DOUBLE PRECISION offset,cstg_bkpt,rmxstge
```

wcaout.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: wcaout.F,v 1.25 2005/06/10 13:29:22 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wcaout.F,v $
C   $Name: rel-5-5 $
C
C  THIS SUBROUTINE IS A MANAGEMENT MODULE THAT DETERMINES THE DISCHARGES AT THE
C  OUTFLOW STRUCTURES OF THE WCAS. DISCHARGES ARE PUT IN KFLO ARRAY.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE wcaout (crmin, wsnde1, wsnde2, wsnde3, wsndes, wsndese,
     & wsneedl8, rlok_to_sa1, wsdel_to_c51w, prevqu, wsdel_to_l8s,
     & total_add_supply_to_lwdd, wsdel_to_l8, jday, rain, volwtr,
     & tinflow, outflow_from_3a, istr_count, avg_sim_wca_stg_floor,
     & outflow_from_sta2_for_ws, istr_count_knflw, s31dsg_to_bay,
     & tempf_volwtr, tempf_weircff, no_of_hurr_pred, struc_name_meas,
     & iopt_lim_des_cap_wca, isdnpflw_for_slope_wca,
     & isuppflw_for_slope_wca, avvol_rlok_to_sa1, avvolws, avvolrg,
     & total_outf_frm_cnl, reg_rel_frm_lok, tneed_pipel,
     & avail_vol_pipel, tneed_orig,total_small_afact,reg_cap_fact,
     & fnm,out_flow_east0)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'et.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
```

```
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'STRCOUNT.inc'
      INCLUDE 'loktarg.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'opersched.inc'
      INCLUDE 'climvar.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION avail_vol_pipel(max_n_eaa_condt),
     & avg_sim_wca_stg_floor(max_n_wcas,max_n_minfllvl_loc),
     & avvolrg(max_n_wcas,max_n_conv_cnl_wca),
     & avvolws(max_n_wcas,max_n_conv_cnl_wca),
     & crmin(max_ncnls,max_n_seas),total_small_afact(max_ncells),
     & env_thru_struc(max_n_ostruc_wca,max_n_wcas),
     & iopt_lim_des_cap_wca(max_n_ostruc_wca,max_n_wcas),
     & isdnpflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
     & isuppflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
     & outflow_from_3a(max_n_conv_cnl_wca), prevqu(max_ncnls),
     & rain(max_ncells), reg_rel_frm_lok(max_n_eaa_condt),
     & reg_thru_struc(max_n_ostruc_wca,max_n_wcas),
     & tempf_volwtr(max_n_res), tempf_weircff(max_n_res,max_nostr_res),
     & tinflow(max_n_wcas,max_n_conv_cnl_wca),
     & tneed_orig(max_n_wcas,max_n_conv_cnl_wca),
     & tneed_pipel(max_n_eaa_condt),
     & total_outf_frm_cnl(max_n_wcas,max_n_conv_cnl_wca),
     & volwtr(max_n_res), ws_thru_struc(max_n_ostruc_wca,max_n_wcas),
     & reg_cap_fact(max_n_ostruc_wca,max_n_wcas,12)

      CHARACTER struc_name_meas(max_n_struc_meas)*6,
     & fnm(max_n_str_node_dep)*6

      LOGICAL lflowws

      DOUBLE PRECISION crmin,hw,tw,stg_lim,stgmin_out
     & ,flshbrd,stgminrg,stgmndn,tw_for_constrnt,tw_s31
     & ,head_drop_s11,hw_s31
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      nodef(ix,iy)=ix-minx(iy)+1+isum(iy)

C  INITIALIZE VARIABLES

      wss31=0.0
      wss151=0.0
      total_outflow_env_2b=0.0
      iconvey_c_s151=iconvey_canal_no(is150bws-6)
      wsctb=1.0
      if (runmode .eq. 'CALIB') THEN
         do isa_c = 1,nwcas
            frac_del_frm_wca(isa_c) = 1.0
         enddo
      endif

      ratflw=1.0
      wss39=0.0
```

```
      regs39=0.0
      tdqu=0.0
      IF (runmode.eq.'SIMUL') THEN
        CALL match ('S333  ', 1, 6, struc_name_meas, nflpts, 6,
     &   imatch_s333)
        CALL match ('S355  ', 1, 6, struc_name_meas, nflpts, 6,
     &   imatch_s355)
        IF (imatch_s333.eq.0) THEN
          kflo(ifls333)=0.0
        END IF
        IF (imatch_s355.eq.0) THEN
          kflo(ifls355)=0.0
        END IF
        kflo(ifls39)=0.0
      END IF
      add_supply_to_ese=0.0
      add_supply_to_e2=0.0
      total_outf_g94s  =0.0
C
      CALL match (name_res_spec_code(13), 1, 6, resname, ntotres+1, 6,
     &imatch)
      ilkbeltc=imatch
      IF (imatch.eq.0) ilkbeltc=ntotres+1
      tvolwtr=volwtr(ilkbeltc)
      DO ir=1,ntotres
        tvolwtr_res(ir)=volwtr(ir)
      END DO
c
c  supplemental demand in LWDD
c
      twsneed=wsnde1+wsnde2+wsnde3+wsndese
      twsneed_lwdd = twsneed
C
C
C  COMPUTE DISCHARGES FOR OUTLET STRUCTURES IN WCAS
C
      istr_count_knflw=istr_count
      DO 50 ica=1,nwcas
        isa=ipwca(ica)
        DO 50 is=1,nstrpt(isa)
        iflo=istrcpt(is,isa)
        imatch_1=0
        DO i=1,ncalcpt
          IF (flnm(iflo).eq.flnm(icalcpt(i))) imatch_1=1
        END DO
        IF (icode_s(is,isa).eq.'GEN'.and.flnm(iflo)
     &   .ne.name_struc_w_spec_code_wca(9)) imatch_1=1
        IF (imatch_1.eq.0) THEN
          istr_count_knflw=istr_count_knflw+1
          istr_index(istr_count_knflw)=iflo
        END IF
        istr_count=istr_count+1
        ws_thru_struc(is,isa)=0.0
C
C  DEFINE THE HEADWATER AND TAILWATER STAGES FOR EACH STRUCTURE
C  OPTIONS IHWOPT(IS,ISA) , ITWOPT(IS,ISA) ARE INCORPORATED
C  SO THE MODEL KNOWS IF THE HEADWATER AND TAILWATER OF A
```

```
C  PARTICULAR STRUCTURE IS REPRESENTED BY A CANAL OR A NODE
C
        IF (ihwopt(is,isa).eq.0) THEN
          ihwlc=icnlhw(is,isa)
          hw=chdep(ihwlc)
        ELSE
          ihwlc=nodef(ixhw(is,isa),iyhw(is,isa))
          dpthgw=(ells(ihwlc)-h(ihwlc))*s(ihwlc)
          IF (pond(ihwlc).gt.dpthgw) THEN
            hw=pond(ihwlc)-dpthgw+ells(ihwlc)
          ELSE
            hw=(pond(ihwlc)-dpthgw)/s(ihwlc)+ells(ihwlc)
          END IF
        END IF
C
        IF (itwopt(is,isa).eq.0) THEN
          itwlc=icnltw(is,isa)
          tw=chdep(itwlc)+hdc(itwlc,iseas)
C
          IF (flnm(istrcpt(is,isa)).eq.name_struc_w_spec_code_wca(17))
     &     THEN
            tw=chdep(itwlc)+hdc(itwlc,iseas)*0.6
          END IF
C
        ELSE
          itwlc=nodef(ixtw(is,isa),iytw(is,isa))
          dpthgw=(ells(itwlc)-h(itwlc))*s(itwlc)
          IF (pond(itwlc).gt.dpthgw) THEN
            tw=pond(itwlc)-dpthgw+ells(itwlc)
          ELSE
            tw=(pond(itwlc)-dpthgw)/s(itwlc)+ells(itwlc)
          END IF
          tw=max(tw,ells(itwlc))
        END IF
        iwzone=iwca_reg_zone(is,isa)
C
        IF (icode_s(is,isa).eq.'GEN') THEN
C
C  GENERAL CODE FOR DISCHARGES
C
          initialize=1
          DO i=1,ncalcpt
            IF (flnm(iflo).eq.flnm(icalcpt(i))) initialize=0
          END DO
          IF (initialize.eq.1.and.flnm(iflo)
     &     .ne.name_struc_w_spec_code_wca(7)) kflo(iflo)=0.0
          IF (flnm(iflo).eq.name_struc_w_spec_code_wca(9)) THEN
            IF (hw.lt.13.0) THEN
              rmaxgo=1.3*(hw-11.0)+1.5
            ELSE
              rmaxgo=hw-8.5
            END IF
            rmaxgo=amax1(rmaxgo,0.0)
C
            IF (hw.gt.tw) THEN
              IF (tw.le.12.1) THEN
                crhds11=0.8235*tw+3.2
```

```
          ELSE
            crhds11=1.71*tw-7.53
          END IF
          IF (hw.le.crhds11) THEN
            rmgos11=5.0*(7.0-rmaxgo)*(crhds11-hw)+rmaxgo
            rmgo=amin1(rmgos11,7.0)
          ELSE
            rmgos11=0.04327*tw**2+8.55*exp(-(hw-tw))+1.56*log(hw-tw)
&            -3.95
            rmgo=amin1(rmaxgo,rmgos11)
          END IF
          rmgo=amax1(rmgo,0.0)
        END IF
      ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(6)) THEN
        IF (hw.gt.tw) THEN
          rmgos10=-490/hw+41.0/(hw-tw)**1.5-0.00166*hw**3+0.00548*
&          tw**2.5+33.42
          rmgo=amin1(rmgos10,6.5)
          rmgo=amax1(rmgo,0.0)
        END IF
      ELSE
        rmgo=1.0
      END IF

      IF (hw.gt.tw) THEN
        struc_cap=dcoeff(is,isa)*rmgo*(hw-tw)**power(is,isa)
      ELSE
        struc_cap=0.0
      END IF
      IF (iopt_lim_des_cap_wca(is,isa).eq.1) THEN
        struc_cap=amin1(struc_cap,desgncp(is,isa))
      END IF
      struc_cap=struc_cap*capfact(is,isa)
      IF ((flnm(iflo).eq.name_struc_w_spec_code_wca(10)
&      .or.flnm(iflo).eq.name_struc_w_spec_code_wca(11)
&      .or.flnm(iflo).eq.name_struc_w_spec_code_wca(12))
&      .and.sim_opt(is,isa).eq.'ENV') THEN
        IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
          tw_for_constrnt=0.0
          DO kk=1,n_cells_c(is,isa)
            tw_for_constrnt=h(igrid_loc_constrnt(is,isa,kk))+
&            pond(igrid_loc_constrnt(is,isa,kk))+tw_for_constrnt
          END DO
          tw_for_constrnt=tw_for_constrnt/n_cells_c(is,isa)
          rmxstge(is,isa,1)=regstg_wca(idn_wca_trig(is,isa),
&          nzone(idn_wca_trig(is,isa)+1)-1)
        ELSE
          tw_for_constrnt=tw
        END IF
        IF (tw_for_constrnt.gt.rmxstge(is,isa,1)-0.25) THEN
          capac_frac_down=4.0*max(rmxstge(is,isa,1)-
&          tw_for_constrnt,0.0)
        ELSE
          capac_frac_down=1.0
        END IF
        struc_cap=struc_cap*capac_frac_down
      END IF
```

```
          IF (flnm(iflo).eq.name_struc_w_spec_code_wca(9)) THEN
            iconv_c=intcnl(idn_wca_indx(is7bsn-6),
     &       iconvey_canal_no(is7bsn-6))
            IF (iconv_c.le.nch) THEN
              IF (iconvey_canal_no(is8bsn-6)
     &         .ne.iconvey_canal_no(is150bws-6)) THEN
               IF (hw.gt.tw) THEN
                 IF (hw-tw.gt.2.5) THEN
                   frac_cap_s11=1.0
                 ELSE
                   frac_cap_s11=(hw-tw)/2.5
                 END IF
                 avvol_above_tw=frac_cap_s11*struc_cap
                 indx_s11=iconv_canal_up_no(is,isa)
                 stg_lim=(hw+tw)/2.0
                avvol_above_twc=(hw-stg_lim)*cl(iconv_c)*width(iconv_c)
     &           +tovflc(iconv_c,stg_lim,0,0.,0,1)+cgseep(iconv_c,
     &           stg_lim,0,0.)+(tinflow(isa,indx_s11)-(kflo(ifls144)+
     &           kflo(ifls145)+kflo(ifls146)))*86400.
                 avvol_above_twc=amax1(avvol_above_twc,0.0)/86400.
                 avvol_above_tw=min(avvol_above_twc,avvol_above_tw)
               ELSE
                 avvol_above_tw=0.0
               END IF
              ELSE
               iconv_c=intcnl(idn_wca_indx(is7bsn-6),
     &          iconvey_canal_no(is7bsn-6))
               indx_s11=iconv_canal_up_no(is,isa)
               stg_lim=(hw+tw)/2.0
               avvol_above_tw=(hw-stg_lim)*cl(iconv_c)*width(iconv_c)+
     &          tovflc(iconv_c,stg_lim,0,0.,0,1)+cgseep(iconv_c,
     &          stg_lim,0,0.)+(tinflow(isa,indx_s11)-(kflo(ifls144)+
     &          kflo(ifls145)+kflo(ifls146)))*86400.
               avvol_above_tw=amax1(avvol_above_tw,0.0)/86400.
              END IF
            ELSE
              avvol_above_tw=999999.
            END IF
          ELSE
            avvol_above_tw=9999999.
          END IF
          IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7)) s39cap=
     &     struc_cap
C
C  if impose regulation schedule,maximum flow is
C  reduced from struc_cap to 0 as trigger stage in  WCA
C  reduces from offset (in ft) above schedule to schedule
C
          IF (sim_opt(is,isa).eq.'REG') THEN
            strtstg=regstg_wca(isa,iwca_reg_zone(is,isa))+offset_reg(is,
     &       isa)
            IF (trig_wca_stg(isa).gt.strtstg) THEN
              struc_flow_max=struc_cap
            ELSE IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)) THEN
              struc_flow_max=(1.0/(strtstg-regstg_wca(isa,iwzone))**2.0)
     &         *struc_cap*(trig_wca_stg(isa)-regstg_wca(isa,iwzone))**
     &         2.0
```

```
            ELSE
              struc_flow_max=0.0
            END IF
          END IF
C
C  determine maximum available volume of water available for outflow
C
          IF (ihwopt(is,isa).eq.0) THEN
C  if headwater is canal stage
            avail_vol=avvolrg(isa,iconv_canal_up_no(is,isa))
            avail_vol_ws=avvolws(isa,iconv_canal_up_no(is,isa))
          ELSE
C  if headwater is nodal stage
            av_vol=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*gdar
            avail_vol=amax1(av_vol,0.0)/86400.
            avail_vol_ws=avail_vol
          END IF
C
C  determine the outflow from structure
C
          IF (itwopt(is,isa).eq.0) THEN
C  tailwater is canal stage
            wsneed=0.0
            DO i=1,no_dsws_canal(is,isa)
              wsneed=wsneed+dqu(idn_ws_cnl(is,isa,i))
            END DO
            IF (nint(wsneed).gt.0) THEN
              IF (ihwopt(is,isa).eq.0) THEN
C
C   downstream needs met via conveyance canal
C
                IF (iprorte(isa).eq.1) THEN
                  ratio=avail_vol_ws/tneed(isa,iconv_canal_up_no(is,isa)
     &             )
                  avvol_upstream=ratio*wsneed
                  ratio=amin1(ratio,1.0)
                  wsdel=wsneed*ratio
                  cout=amin1(wsdel,struc_cap)*frac_del_frm_wca(isa)
                  IF (flnm(iflo).eq.name_struc_w_spec_code_wca(13))
     &             THEN
                    rem_deliv_for_nnrc=(1.0-ratio)*dqu(innrc)
                  END IF
                  ratio1 = cout/wsneed
                  tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &             iconv_canal_up_no(is,isa))-wsneed
                  DO i=1,no_dsws_canal(is,isa)
                    qu(idn_ws_cnl(is,isa,i))=qu(idn_ws_cnl(is,isa,i))+
     &               ratio1*dqu(idn_ws_cnl(is,isa,i))*86400.
c                   if (flnm(iflo).eq.'S38  ') print *,wsneed
c    &               ,dqu(idn_ws_cnl(is,isa,i)),ratio,ratio1
c    &               ,cout,frac_del_frm_wca(isa),'  S38WS'
                    dqu(idn_ws_cnl(is,isa,i))=(1.0-ratio1)*
     &               dqu(idn_ws_cnl(is,isa,i))
                    IF (ivary_slope_daily_opt(idn_ws_cnl(is,isa,i))
     &               .eq.1) THEN
                      CALL match (cnm(idn_ws_cnl(is,isa,i)), 1, 5,
     &                 canal_name_dvslope, ncnl_dly_slope, 5, imatch)
```

```
                  sgn=1
                  IF (isuppflw_for_slope_wca(is,isa).eq.3) THEN
                    sgn=-1
                  END IF
                  dischg_for_slope_kn(imatch,1,
&                  isdnpflw_for_slope_wca(is,isa))=
&                  dischg_for_slope_kn(imatch,1,
&                  isdnpflw_for_slope_wca(is,isa))+sgn*ratio1*
&                  dqu(idn_ws_cnl(is,isa,i))
                END IF
              END DO
              total_cout=cout
              avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
&              iconv_canal_up_no(is,isa))-cout
              avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
&              iconv_canal_up_no(is,isa))-cout
              avvolrg(isa,iconv_canal_up_no(is,isa))=
&              amax1(avvolrg(isa,iconv_canal_up_no(is,isa)),0.0)
              avail_vol_ws=avvolws(isa,iconv_canal_up_no(is,isa))
            ELSE
              avvol_upstream=avail_vol_ws
              total_cout=0.0
              DO i=1,no_dsws_canal(is,isa)
                cout=amin1(struc_cap,dqu(idn_ws_cnl(is,isa,i)),
&                avvolws(isa,iconv_canal_up_no(is,isa)))*
&                frac_del_frm_wca(isa)
                qu(idn_ws_cnl(is,isa,i))=qu(idn_ws_cnl(is,isa,i))+
&                cout*86400.
                IF (ivary_slope_daily_opt(idn_ws_cnl(is,isa,i))
&                .eq.1) THEN
                  CALL match (cnm(idn_ws_cnl(is,isa,i)), 1, 5,
&                  canal_name_dvslope, ncnl_dly_slope, 5, imatch)
                  sgn=1
                  IF (isuppflw_for_slope_wca(is,isa).eq.3) THEN
                    sgn=-1
                  END IF
                  dischg_for_slope_kn(imatch,1,
&                  isdnpflw_for_slope_wca(is,isa))=
&                  dischg_for_slope_kn(imatch,1,
&                  isdnpflw_for_slope_wca(is,isa))+sgn*cout
                END IF
                upstream_inflow(idn_ws_cnl(is,isa,i))=
&                upstream_inflow(idn_ws_cnl(is,isa,i))+cout*86400.
                IF (cnm(idn_ws_cnl(is,isa,i)).eq.'C13  ') THEN
                  rem_deliv_for_nnrc=cout-dqu(idn_ws_cnl(is,isa,i))+
&                  dqu(innrc)
                  rem_deliv_for_nnrc=amax1(rem_deliv_for_nnrc,0.0)
                END IF
                dqu(idn_ws_cnl(is,isa,i))=dqu(idn_ws_cnl(is,isa,i))-
&                cout
                struc_cap=struc_cap-cout
                avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
&                iconv_canal_up_no(is,isa))-cout
                avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
&                iconv_canal_up_no(is,isa))-cout
                avvolrg(isa,iconv_canal_up_no(is,isa))=
&                amax1(avvolrg(isa,iconv_canal_up_no(is,isa)),0.0)
```

```
                  total_cout=total_cout+cout
                END DO
                avail_vol_ws=avvolws(isa,iconv_canal_up_no(is,isa))
              END IF
            ELSE
C
C  downstream needs met from grid cell
C
              IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &  ') THEN
                avvol_upstream=avail_vol_ws
                total_cout=amin1(struc_cap,wsneed,avail_vol_ws)*
     &           frac_del_frm_wca(isa)
                ratio_dist=total_cout/wsneed
                IF (flnm(iflo).eq.name_struc_w_spec_code_wca(13))
     &           THEN
                  rem_deliv_for_nnrc=(1.0-ratio_dist)*dqu(innrc)*
     &             frac_del_frm_wca(isa)
                END IF
              ELSE
                avvol_upstream=avail_vol_pipel(iserv_area_indx(is,isa)
     &           )
                ratio_dist=avvol_upstream/
     &           tneed_pipel(iserv_area_indx(is,isa))
                total_cout=ratio_dist*wsneed*frac_del_frm_wca(isa)
              END IF
              DO i=1,no_dsws_canal(is,isa)
                qu(idn_ws_cnl(is,isa,i))=qu(idn_ws_cnl(is,isa,i))+
     &           ratio_dist*dqu(idn_ws_cnl(is,isa,i))*86400.
                IF (ivary_slope_daily_opt(idn_ws_cnl(is,isa,i)).eq.1)
     &           THEN
                  CALL match (cnm(idn_ws_cnl(is,isa,i)), 1, 5,
     &             canal_name_dvslope, ncnl_dly_slope, 5, imatch)
                  sgn=1
                  IF (isuppflw_for_slope_wca(is,isa).eq.3) THEN
                    sgn=-1
                  END IF
                  dischg_for_slope_kn(imatch,1,
     &             isdnpflw_for_slope_wca(is,isa))=
     &             dischg_for_slope_kn(imatch,1,
     &             isdnpflw_for_slope_wca(is,isa))+sgn*ratio_dist*
     &             dqu(idn_ws_cnl(is,isa,i))
                END IF
                upstream_inflow(idn_ws_cnl(is,isa,i))=
     &           upstream_inflow(idn_ws_cnl(is,isa,i))+ratio_dist*
     &           dqu(idn_ws_cnl(is,isa,i))*86400.
                IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).eq.'PIP
     &EL  ') THEN
                  tneed_pipel(iserv_area_indx(is,isa))=
     &             tneed_pipel(iserv_area_indx(is,isa))-
     &             dqu(idn_ws_cnl(is,isa,i))
                END IF
                dqu(idn_ws_cnl(is,isa,i))=(1.0-ratio_dist)*
     &           dqu(idn_ws_cnl(is,isa,i))
              END DO
              IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).eq.'PIPEL
     &  ') THEN
```

```
                avail_vol_pipel(iserv_area_indx(is,isa))=
     &           avail_vol_pipel(iserv_area_indx(is,isa))-total_cout
              END IF
            END IF
            struc_flow=total_cout
            ws_thru_struc(is,isa)=struc_flow
          END IF
C
C excess flow from water conservation area
C
          avail_vol=amax1(avail_vol-ws_thru_struc(is,isa),0.0)
          avvol_upstream=avail_vol
          IF (sim_opt(is,isa).eq.'ENV') THEN
            itarg=0
            DO i=1,no_of_targets(is,isa)
              IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                itarg=itarg+1
              END IF
            END DO
            iup_targ=iup_targ_indx_wca(is,isa)
            r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
            r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
            IF (iup_targ.gt.ntotal_no_import_areas) THEN
              r1top=999.
              r1bot=998.
            END IF
            iconv_c=iconv_canal_up_no(is,isa)
            IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
              IF (ihwopt(is,isa).eq.0) THEN
                stgmin_out=r1top-0.2
                cnlstrg=(chdep(intcnl(isa,iconv_c))-stgmin_out)*
     &           cl(intcnl(isa,iconv_c))*width(intcnl(isa,iconv_c))
                avail_vol_ws=cnlstrg+cgseep(intcnl(isa,iconv_c),
     &           stgmin_out,0,0.)+tovflc(intcnl(isa,iconv_c),
     &           stgmin_out,0,0.,0,1)+tinflow(isa,iconv_c)*86400.-
     &           ws_thru_struc(is,isa)*86400.
                avail_vol_ws=amax1(avail_vol_ws,0.0)/86400.
                avail_vol_ws=avail_vol_ws-total_outf_frm_cnl(isa,
     &           iconv_c)
                avail_vol_ws=amax1(avail_vol_ws,0.0)
              END IF
              IF (itarg.eq.0) THEN
                IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                  IF ((flnm(iflo).ne.name_struc_w_spec_code_wca(10)
     &             .and.flnm(iflo).ne.name_struc_w_spec_code_wca(11)
     &             .and.flnm(iflo).ne.name_struc_w_spec_code_wca(12))
     &             .or.no_of_targets(is,isa).gt.0) THEN
                    struc_outflow=2.0*struc_cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1top)
                    struc_outflow=amin1(struc_outflow,struc_cap)
                    reg_thru_struc(is,isa)=amin1(struc_outflow,
     &               avvol_above_tw,avail_vol_ws)
                    env_thru_struc(is,isa)=0.0
                  ELSE
                    struc_outflow=struc_cap
                    reg_thru_struc(is,isa)=amin1(struc_outflow,
     &               avvol_above_tw,avail_vol_ws)
```

```
                  env_thru_struc(is,isa)=0.0
                END IF
              ELSE
                struc_outflow=0.0
                reg_thru_struc(is,isa)=0.0
                env_thru_struc(is,isa)=0.0
              END IF
            ELSE
              struc_max1=2.0*struc_cap*
&              (avg_sim_wca_stg_out(iup_targ)-r1top)
              struc_max1=amin1(struc_max1,struc_cap)
              notarg=0
              total_dpth_below_targ=0.0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  notarg=notarg+1
                  dpth_below_targ=
&                  stage_import_wca(itarg_indx_wca(is,isa,i))-
&                  offset(itarg_indx_wca(is,isa,i),1)-
&                  avg_sim_wca_stg(itarg_indx_wca(is,isa,i))
                  dpth_below_targ=amax1(dpth_below_targ,0.0)
                  total_dpth_below_targ=dpth_below_targ+
&                  total_dpth_below_targ
                END IF
              END DO
              avg_dpth_below_targ=total_dpth_below_targ/notarg
              IF (avg_dpth_below_targ.lt.offset_reg(is,isa)) THEN
                struc_outflow=(1.0/offset_reg(is,isa))*
&                avg_dpth_below_targ*(struc_cap-struc_max1)+
&                struc_max1
              ELSE
                struc_outflow=struc_cap
              END IF
              reg_thru_struc(is,isa)=0.0
              env_thru_struc(is,isa)=amin1(struc_outflow,
&              avail_vol_ws,avvol_above_tw)
            END IF
          ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
            IF (ihwopt(is,isa).eq.0) THEN
              stgmin_out=r1bot-0.1
              cnlstrg=(chdep(intcnl(isa,iconv_c))-stgmin_out)*
&              cl(intcnl(isa,iconv_c))*width(intcnl(isa,iconv_c))
              avail_vol_ws=cnlstrg+cgseep(intcnl(isa,iconv_c),
&              stgmin_out,0,0.)+tovflc(intcnl(isa,iconv_c),
&              stgmin_out,0,0.,0,1)+tinflow(isa,iconv_c)*86400.-
&              ws_thru_struc(is,isa)*86400.
              avail_vol_ws=amax1(avail_vol_ws,0.0)/86400.
              avail_vol_ws=avail_vol_ws-total_outf_frm_cnl(isa,
&              iconv_c)
              avail_vol_ws=amax1(avail_vol_ws,0.0)
            END IF
            IF (itarg.gt.0.or.((flnm(iflo)
&            .eq.name_struc_w_spec_code_wca(10).or.flnm(iflo)
&            .eq.name_struc_w_spec_code_wca(11).or.flnm(iflo)
&            .eq.name_struc_w_spec_code_wca(12))
&            .and.no_of_targets(is,isa).eq.0)) THEN
              struc_outflow=1/(r1top-r1bot)*struc_cap*
```

```
&             (avg_sim_wca_stg_out(iup_targ)-r1bot)
            IF ((flnm(iflo).eq.name_struc_w_spec_code_wca(10)
&            .or.flnm(iflo).eq.name_struc_w_spec_code_wca(11)
&            .or.flnm(iflo).eq.name_struc_w_spec_code_wca(12))
&            .and.no_of_targets(is,isa).eq.0) THEN
              reg_thru_struc(is,isa)=amin1(struc_outflow,
&              avvol_above_tw,avail_vol_ws)
              env_thru_struc(is,isa)=0.0
            ELSE
              notarg=0
              total_dpth_below_targ=0.0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  notarg=notarg+1
                  dpth_below_targ=
&                  stage_import_wca(itarg_indx_wca(is,isa,i))-
&                  offset(itarg_indx_wca(is,isa,i),1)-
&                  avg_sim_wca_stg(itarg_indx_wca(is,isa,i))
                  dpth_below_targ=amax1(dpth_below_targ,0.0)
                  total_dpth_below_targ=dpth_below_targ+
&                  total_dpth_below_targ
                END IF
              END DO
              avg_dpth_below_targ=total_dpth_below_targ/notarg
              IF (avg_dpth_below_targ.lt.offset_reg(is,isa)) THEN
                struc_outflow1=(1.0/offset_reg(is,isa))*
&                avg_dpth_below_targ*struc_cap
              ELSE
                struc_outflow1=struc_cap
              END IF
              struc_outflow=amin1(struc_outflow,struc_outflow1)
              reg_thru_struc(is,isa)=0.0
              env_thru_struc(is,isa)=amin1(struc_outflow,
&              avail_vol_ws,avvol_above_tw)
            END IF
          ELSE
            struc_outflow=0.0
            reg_thru_struc(is,isa)=0.0
            env_thru_struc(is,isa)=0.0
          END IF
        ELSE
          struc_outflow=0.0
          reg_thru_struc(is,isa)=0.0
          env_thru_struc(is,isa)=0.0
        END IF
        avvol_upstream=avail_vol_ws
        struc_flow=amin1(struc_outflow,avvol_above_tw,
&        avail_vol_ws)
        IF (ihwopt(is,isa).eq.0) THEN
          env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
&           avvolws(isa,iconv_canal_up_no(is,isa)))
          reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
&           avvolws(isa,iconv_canal_up_no(is,isa)))
          struc_flow=amin1(struc_flow,avvolws(isa,
&           iconv_canal_up_no(is,isa)))
          total_outf_frm_cnl(isa,iconv_c)=total_outf_frm_cnl(isa,
&           iconv_c)+struc_flow
```

```
              END IF
            ELSE IF (sim_opt(is,isa).eq.'REG') THEN
              IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
                tw_for_constrnt=0.0
                DO kk=1,n_cells_c(is,isa)
                  tw_for_constrnt=h(igrid_loc_constrnt(is,isa,kk))+
     &             pond(igrid_loc_constrnt(is,isa,kk))+tw_for_constrnt
                END DO
                tw_for_constrnt=tw_for_constrnt/n_cells_c(is,isa)
                rmxstge(is,isa,1)=regstg_wca(idn_wca_trig(is,isa),
     &           nzone(idn_wca_trig(is,isa)+1)-1)
              ELSE
                tw_for_constrnt=tw
              END IF
              IF (((trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+
     &         1.0.or.(trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
     &         .or.iopt_for_lokreg_flwth(is,isa).eq.1))
     &         .and.tw_for_constrnt.lt.rmxstge(is,isa,1)+0.50)
     &         .and.iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
                   IF (tw_for_constrnt.gt.rmxstge(is,isa,1)-0.25) THEN
                     capac_frac_down=4.0*max(rmxstge(is,isa,1)-
     &               tw_for_constrnt,0.0)
                   ELSE
                     capac_frac_down=1.0
                   END IF
                ELSE
                   IF (tw_for_constrnt.gt.rmxstge(is,isa,1)-0.50) THEN
                     capac_frac_down=1.00*max(rmxstge(is,isa,1)+0.5-
     &               tw_for_constrnt,0.0)
                   ELSE
                     capac_frac_down=1.0
                   END IF
                ENDIF
                IF (iopt_for_lokreg_flwth(is,isa).eq.0) THEN
                  IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+1.0)
     &             THEN
                    IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+
     &               1.25) THEN
                      capac_frac_up=1.0
                    ELSE
                      capac_frac_up=4.0*(trig_wca_stg(isa)-
     &                 regstg_wca(isa,iwzone)-1.0)
                    END IF
                  END IF
                ELSE
                  capac_frac_up=1.0
                END IF
                capac_frac_up_fc=amax1(capac_frac_down,capac_frac_up)
                IF (flnm(iflo).eq.'S144'.or.flnm(iflo).eq.'S145'
     &           .or.flnm(iflo).eq.'S146') THEN
                  capac_frac_up_fc=capac_frac_down
                END IF
C               if (flnm(iflo).eq.'S144') print *,TW_FOR_CONSTRNT
C    $               ,rmxstge(is,isa,1),regstg_wca(isa,iwzone)
C    $               ,trig_wca_stg(isa),capac_frac_up,capac_frac_down
C    $               ,capac_frac_up_fc
```

```
                IF (rmulti_seas_pred(month)
     &           .gt.rmulti_seas_thres_struc(is,isa)
     &           .or.iatl_convyr_index.eq.1) THEN
                  struc_flow_max=struc_flow_max-ws_thru_struc(is,isa)
                  struc_flow_max=amax1(struc_flow_max,0.0)
     &            * reg_cap_fact(is,isa,month)
c                 print *,is,isa,flnm(iflo),reg_cap_fact(is,isa,month)
                  IF (iopt_for_lokreg_flwth(is,isa)
     &             .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,iwzone))
     &              THEN
                    IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+1.0)
     &                THEN
                      struc_flow=amin1(struc_flow_max,avail_vol,
     &                 avvol_above_tw)*capac_frac_up_fc
                      IF (iopt_for_lokreg_flwth(is,isa).eq.1)
     &                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
     &                 avail_vol_ws)*capac_frac_up_fc
                    ELSE
                      struc_flow=amin1(struc_flow_max,avail_vol,
     &                 avvol_above_tw)*capac_frac_down
                      IF (iopt_for_lokreg_flwth(is,isa).eq.1)
     &                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
     &                 avail_vol_ws)*capac_frac_down
                    END IF
                    IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7))
     &               THEN
                      IF (no_puls_zones.gt.0) THEN
                        CALL bpts (iday, month, 10.0, 1, mzone_puls_bot,
     &                    1, stgref_lok_sched_bot)
                      ELSE
                        CALL bpts (iday, month, 10.0, 1, n_fc_zones, 1,
     &                   stgref_lok_sched_bot)
                      END IF
                      IF (trig_wca_stg(isa).lt.regstg_wca(isa,
     &                 itop_zone_indx(isa+1))
     &                 .and.stagelo.le.stgref_lok_sched_bot+0.001)
     &                 struc_flow=0.0
                      IF (iopt_for_lokreg_flwth(is,isa).eq.1)
     &                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
     &                 avail_vol_ws)
                    END IF
                  ELSE
                    struc_flow=amin1(struc_flow_max,avvol_above_tw,
     &               reg_rel_frm_lok(ieaa_conduit_indx(is,isa)),
     &               avail_vol_ws)
                    reg_rel_frm_lok(ieaa_conduit_indx(is,isa))=
     &               reg_rel_frm_lok(ieaa_conduit_indx(is,isa))-
     &               struc_flow
                  END IF
                  reg_thru_struc(is,isa)=struc_flow
                  env_thru_struc(is,isa)=0.0
                ELSE
                  struc_flow=0.0
                  env_thru_struc(is,isa)=0.0
                  reg_thru_struc(is,isa)=0.0
                END IF
              ELSE
```

```
                struc_flow=0.0
                env_thru_struc(is,isa)=0.0
                reg_thru_struc(is,isa)=0.0
              END IF
            END IF
            IF (idn_reserv_no(is,isa).le.ntotres) THEN
              IF (lcnb(icnltw(is,isa)).eq.ibsn_no_res(idn_reserv_no(is,
     &         isa))) THEN
                struc_flow=amin1(struc_flow,volwtr(idn_reserv_no(is,isa)
     &           ))
                env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
     &           volwtr(idn_reserv_no(is,isa)))
                reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
     &           volwtr(idn_reserv_no(is,isa)))
                volwtr(idn_reserv_no(is,isa))=volwtr(idn_reserv_no(is,
     &           isa))-struc_flow
              ELSE
                struc_flow=amin1(struc_flow,
     &           tvolwtr_res(idn_reserv_no(is,isa)))
                env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
     &           tvolwtr_res(idn_reserv_no(is,isa)))
                reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
     &           tvolwtr_res(idn_reserv_no(is,isa)))
              END IF
              tvolwtr_res(idn_reserv_no(is,isa))=
     &         tvolwtr_res(idn_reserv_no(is,isa))-struc_flow
            END IF
            IF (ihwopt(is,isa).eq.0) THEN
              avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
     &         iconv_canal_up_no(is,isa))-struc_flow
              avvolrg(isa,iconv_canal_up_no(is,isa))=amax1(avvolrg(isa,
     &         iconv_canal_up_no(is,isa)),0.0)
              avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &         iconv_canal_up_no(is,isa))-struc_flow
            END IF
            IF (flnm(iflo).ne.name_struc_w_spec_code_wca(9)) THEN
              qu(itwlc)=qu(itwlc)+struc_flow*86400.
              upstream_inflow(itwlc)=upstream_inflow(itwlc)+struc_flow*
     &         86400.
              IF (ivary_slope_daily_opt(itwlc).eq.1) THEN
                CALL match (cnm(itwlc), 1, 5, canal_name_dvslope,
     &           ncnl_dly_slope, 5, imatch)
                sgn=1
                IF (isuppflw_for_slope_wca(is,isa).eq.3) THEN
                  sgn=-1
                END IF
                dischg_for_slope_kn(imatch,1,isdnpflw_for_slope_wca(is,
     &           isa))=dischg_for_slope_kn(imatch,1,
     &           isdnpflw_for_slope_wca(is,isa))+sgn*struc_flow
              END IF
            END IF
          ELSE
C  if tailwater is grid cell stage
            wsneed=0.0
            avvol_upstream=avail_vol
            IF (sim_opt(is,isa).eq.'ENV') THEN
              itarg=0
```

```
          DO i=1,no_of_targets(is,isa)
            IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
              itarg=itarg+1
            END IF
          END DO
          iup_targ=iup_targ_indx_wca(is,isa)
          r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
          r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
          IF (iup_targ.gt.ntotal_no_import_areas) THEN
            r1top=999.
            r1bot=998.
          END IF
          iconv_c=iconv_canal_up_no(is,isa)
          IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
            IF (ihwopt(is,isa).eq.0) THEN
              stgmin_out=r1top-0.2
              cnlstrg=(chdep(intcnl(isa,iconv_c))-stgmin_out)*
&              cl(intcnl(isa,iconv_c))*width(intcnl(isa,iconv_c))
              avail_vol_ws=cnlstrg+cgseep(intcnl(isa,iconv_c),
&              stgmin_out,0,0.)+tovflc(intcnl(isa,iconv_c),
&              stgmin_out,0,0.,0,1)+tinflow(isa,iconv_c)*86400.
              avail_vol_ws=amax1(avail_vol_ws,0.0)/86400.
              avail_vol_ws=avail_vol_ws-total_outf_frm_cnl(isa,
&              iconv_c)
              avail_vol_ws=amax1(avail_vol_ws,0.0)
            END IF
            IF (itarg.eq.0) THEN
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                IF ((flnm(iflo).ne.name_struc_w_spec_code_wca(10)
&                .and.flnm(iflo).ne.name_struc_w_spec_code_wca(11)
&                .and.flnm(iflo).ne.name_struc_w_spec_code_wca(12))
&                .or.no_of_targets(is,isa).gt.0) THEN
                  struc_outflow=2.0*struc_cap*
&                  (avg_sim_wca_stg_out(iup_targ)-r1top)
                  struc_outflow=amin1(struc_outflow,struc_cap)
                  IF ((flnm(iflo).eq.name_struc_w_spec_code_wca(10)
&                  .or.flnm(iflo).eq.name_struc_w_spec_code_wca(11)
&                  .or.flnm(iflo).eq.name_struc_w_spec_code_wca(12))
&                  .and.no_of_targets(is,isa).eq.0) THEN
                    reg_thru_struc(is,isa)=amin1(struc_outflow,
&                    avvol_above_tw,avail_vol_ws)
                    env_thru_struc(is,isa)=0.0
                  ELSE
                    reg_thru_struc(is,isa)=0.0
                    env_thru_struc(is,isa)=amin1(struc_outflow,
&                    avvol_above_tw,avail_vol_ws)
                  END IF
                ELSE
                  struc_outflow=struc_cap
                  reg_thru_struc(is,isa)=amin1(struc_outflow,
&                  avvol_above_tw,avail_vol_ws)
                  env_thru_struc(is,isa)=0.0
                END IF
              ELSE
                struc_outflow=0.0
                reg_thru_struc(is,isa)=0.0
                env_thru_struc(is,isa)=0.0
```

```
              END IF
            ELSE
              struc_max1=2.0*struc_cap*
&              (avg_sim_wca_stg_out(iup_targ)-r1top)
              struc_max1=amin1(struc_max1,struc_cap)
              notarg=0
              total_dpth_below_targ=0.0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  notarg=notarg+1
                  dpth_below_targ=
&                  stage_import_wca(itarg_indx_wca(is,isa,i))-
&                  offset(itarg_indx_wca(is,isa,i),1)-
&                  avg_sim_wca_stg(itarg_indx_wca(is,isa,i))
                  dpth_below_targ=amax1(dpth_below_targ,0.0)
                  total_dpth_below_targ=dpth_below_targ+
&                  total_dpth_below_targ
                END IF
              END DO
              avg_dpth_below_targ=total_dpth_below_targ/notarg
              IF (avg_dpth_below_targ.lt.offset_reg(is,isa)) THEN
                struc_outflow=(1.0/offset_reg(is,isa))*
&                avg_dpth_below_targ*(struc_cap-struc_max1)+
&                struc_max1
              ELSE
                struc_outflow=struc_cap
              END IF
              reg_thru_struc(is,isa)=0.0
              env_thru_struc(is,isa)=amin1(struc_outflow,
&              avvol_above_tw,avail_vol_ws)
            END IF
          ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
            IF (ihwopt(is,isa).eq.0) THEN
              stgmin_out=r1bot-0.1
              cnlstrg=(chdep(intcnl(isa,iconv_c))-stgmin_out)*
&              cl(intcnl(isa,iconv_c))*width(intcnl(isa,iconv_c))
              avail_vol_ws=cnlstrg+cgseep(intcnl(isa,iconv_c),
&              stgmin_out,0,0.)+tovflc(intcnl(isa,iconv_c),
&              stgmin_out,0,0.,0,1)+tinflow(isa,iconv_c)*86400.
              avail_vol_ws=amax1(avail_vol_ws,0.0)/86400.
              avail_vol_ws=avail_vol_ws-total_outf_frm_cnl(isa,
&              iconv_c)
              avail_vol_ws=amax1(avail_vol_ws,0.0)
            END IF
            IF (itarg.gt.0.or.((flnm(iflo)
&             .eq.name_struc_w_spec_code_wca(10).or.flnm(iflo)
&             .eq.name_struc_w_spec_code_wca(11).or.flnm(iflo)
&             .eq.name_struc_w_spec_code_wca(12))
&             .and.no_of_targets(is,isa).eq.0)) THEN
              struc_outflow=1/(r1top-r1bot)*struc_cap*
&              (avg_sim_wca_stg_out(iup_targ)-r1bot)
              IF ((flnm(iflo).eq.name_struc_w_spec_code_wca(10)
&               .or.flnm(iflo).eq.name_struc_w_spec_code_wca(11)
&               .or.flnm(iflo).eq.name_struc_w_spec_code_wca(12))
&               .and.no_of_targets(is,isa).eq.0) THEN
                env_thru_struc(is,isa)=0.0
                reg_thru_struc(is,isa)=amin1(struc_outflow,
```

```
&               avvol_above_tw,avail_vol_ws)
              ELSE
                notarg=0
                total_dpth_below_targ=0.0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    notarg=notarg+1
                    dpth_below_targ=
&                    stage_import_wca(itarg_indx_wca(is,isa,i))-
&                    offset(itarg_indx_wca(is,isa,i),1)-
&                    avg_sim_wca_stg(itarg_indx_wca(is,isa,i))
                    dpth_below_targ=amax1(dpth_below_targ,0.0)
                    total_dpth_below_targ=dpth_below_targ+
&                    total_dpth_below_targ
                  END IF
                END DO
                avg_dpth_below_targ=total_dpth_below_targ/notarg
                IF (avg_dpth_below_targ.lt.offset_reg(is,isa)) THEN
                  struc_outflow1=(1.0/offset_reg(is,isa))*
&                  avg_dpth_below_targ*struc_cap
                ELSE
                  struc_outflow1=struc_cap
                END IF
                struc_outflow=amin1(struc_outflow,struc_outflow1)
                reg_thru_struc(is,isa)=0.0
                env_thru_struc(is,isa)=amin1(struc_outflow,
&               avvol_above_tw,avail_vol_ws)
              END IF
            ELSE
              struc_outflow=0.0
              reg_thru_struc(is,isa)=0.0
              env_thru_struc(is,isa)=0.0
            END IF
          ELSE
            struc_outflow=0.0
            reg_thru_struc(is,isa)=0.0
            env_thru_struc(is,isa)=0.0
          END IF
          avvol_upstream=avail_vol_ws
          struc_flow=amin1(struc_outflow,avvol_above_tw,
&         avail_vol_ws)
          IF (ihwopt(is,isa).eq.0) THEN
            env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
&            avvolws(isa,iconv_canal_up_no(is,isa)))
            reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
&            avvolws(isa,iconv_canal_up_no(is,isa)))
            struc_flow=amin1(struc_flow,avvolws(isa,
&            iconv_canal_up_no(is,isa)))
            total_outf_frm_cnl(isa,iconv_c)=total_outf_frm_cnl(isa,
&            iconv_c)+struc_flow
          END IF
        ELSE IF (sim_opt(is,isa).eq.'REG') THEN
          IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
            tw_for_constrnt=0.0
            DO kk=1,n_cells_c(is,isa)
              tw_for_constrnt=h(igrid_loc_constrnt(is,isa,kk))+
&              pond(igrid_loc_constrnt(is,isa,kk))+tw_for_constrnt
```

```
                END DO
                tw_for_constrnt=tw_for_constrnt/n_cells_c(is,isa)
                rmxstge(is,isa,1)=regstg_wca(idn_wca_trig(is,isa),
     &           nzone(idn_wca_trig(is,isa)+1)-1)
              ELSE
                tw_for_constrnt=tw
              END IF
              IF (((trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+
     &         1.0.or.(trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
     &         .or.iopt_for_lokreg_flwth(is,isa).eq.1))
     &         .and.tw_for_constrnt.lt.rmxstge(is,isa,1)+0.50)
     &         .and.iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
                   IF (tw_for_constrnt.gt.rmxstge(is,isa,1)-0.25) THEN
                     capac_frac_down=4.0*max(rmxstge(is,isa,1)-
     &               tw_for_constrnt,0.0)
                   ELSE
                     capac_frac_down=1.0
                   END IF
                ELSE
                   IF (tw_for_constrnt.gt.rmxstge(is,isa,1)-0.50) THEN
                     capac_frac_down=1.00*max(rmxstge(is,isa,1)+0.5-
     &               tw_for_constrnt,0.0)
                   ELSE
                     capac_frac_down=1.0
                   END IF
                ENDIF
                IF (iopt_for_lokreg_flwth(is,isa).eq.0) THEN
                  IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+1.0)
     &             THEN
                    IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+
     &               1.25) THEN
                      capac_frac_up=1.0
                    ELSE
                      capac_frac_up=4.0*(trig_wca_stg(isa)-
     &                 regstg_wca(isa,iwzone)-1.0)
                    END IF
                  END IF
                ELSE
                  capac_frac_up=1.0
                END IF
                capac_frac_up_fc=amax1(capac_frac_down,capac_frac_up)
                IF (flnm(iflo).eq.'S144'.or.flnm(iflo).eq.'S145'
     &           .or.flnm(iflo).eq.'S146') THEN
                  capac_frac_up_fc=capac_frac_down
                END IF
C               if (flnm(iflo).eq.'S144') print *,TW_FOR_CONSTRNT
C    $               ,rmxstge(is,isa,1),regstg_wca(isa,iwzone)
C    $               ,trig_wca_stg(isa),capac_frac_up,capac_frac_down
C    $               ,capac_frac_up_fc
                IF (rmulti_seas_pred(month)
     &           .gt.rmulti_seas_thres_struc(is,isa)
     &           .or.iatl_convyr_index.eq.1) THEN
                  struc_flow_max=struc_flow_max-ws_thru_struc(is,isa)
                  struc_flow_max=amax1(struc_flow_max,0.0)
     &            * reg_cap_fact(is,isa,month)
c                 print *,is,isa,flnm(iflo),reg_cap_fact(is,isa,month)
```

```
              IF (iopt_for_lokreg_flwth(is,isa)
&              .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,iwzone))
&              THEN
               IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)+1.0)
&                THEN
                 struc_flow=amin1(struc_flow_max,avail_vol,
&                 avvol_above_tw)*capac_frac_up_fc
                 IF (iopt_for_lokreg_flwth(is,isa).eq.1)
&                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
&                 avail_vol_ws)*capac_frac_up_fc
               ELSE
                 struc_flow=amin1(struc_flow_max,avail_vol,
&                 avvol_above_tw)*capac_frac_down
                 IF (iopt_for_lokreg_flwth(is,isa).eq.1)
&                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
&                 avail_vol_ws)*capac_frac_down
               END IF
               IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7))
&               THEN
                 IF (no_puls_zones.gt.0) THEN
                   CALL bpts (iday, month, 10.0, 1, mzone_puls_bot,
&                    1, stgref_lok_sched_bot)
                 ELSE
                   CALL bpts (iday, month, 10.0, 1, n_fc_zones, 1,
&                   stgref_lok_sched_bot)
                 END IF
                 IF (trig_wca_stg(isa).lt.regstg_wca(isa,
&                 itop_zone_indx(isa+1))
&                 .and.stagelo.le.stgref_lok_sched_bot+0.01)
&                 struc_flow=0.0
                 IF (iopt_for_lokreg_flwth(is,isa).eq.1)
&                 struc_flow=amin1(struc_flow_max,avvol_above_tw,
&                 avail_vol_ws)
               END IF
              ELSE
               struc_flow=amin1(struc_flow_max,avvol_above_tw,
&               reg_rel_frm_lok(ieaa_conduit_indx(is,isa)),
&               avail_vol_ws)
               reg_rel_frm_lok(ieaa_conduit_indx(is,isa))=
&               reg_rel_frm_lok(ieaa_conduit_indx(is,isa))-
&               struc_flow
              END IF
              reg_thru_struc(is,isa)=struc_flow
              env_thru_struc(is,isa)=0.0
            ELSE
              struc_flow=0.0
              env_thru_struc(is,isa)=0.0
              reg_thru_struc(is,isa)=0.0
            END IF
          ELSE
            struc_flow=0.0
            env_thru_struc(is,isa)=0.0
            reg_thru_struc(is,isa)=0.0
          END IF
        END IF
        IF (idn_reserv_no(is,isa).le.ntotres) THEN
          IF (ibsn(icnltw(is,isa)).eq.ibsn_no_res(idn_reserv_no(is,
```

```
&         isa))) THEN
           struc_flow=amin1(struc_flow,volwtr(idn_reserv_no(is,isa)
&           ))
           env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
&           volwtr(idn_reserv_no(is,isa)))
           reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
&           volwtr(idn_reserv_no(is,isa)))
           volwtr(idn_reserv_no(is,isa))=volwtr(idn_reserv_no(is,
&           isa))-struc_flow
         ELSE
           struc_flow=amin1(struc_flow,
&           tvolwtr_res(idn_reserv_no(is,isa)))
           env_thru_struc(is,isa)=amin1(env_thru_struc(is,isa),
&           tvolwtr_res(idn_reserv_no(is,isa)))
           reg_thru_struc(is,isa)=amin1(reg_thru_struc(is,isa),
&           tvolwtr_res(idn_reserv_no(is,isa)))
         END IF
         tvolwtr_res(idn_reserv_no(is,isa))=
&         tvolwtr_res(idn_reserv_no(is,isa))-struc_flow
       END IF
       IF (ihwopt(is,isa).eq.0) THEN
         avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
&         iconv_canal_up_no(is,isa))-struc_flow
         avvolrg(isa,iconv_canal_up_no(is,isa))=amax1(avvolrg(isa,
&         iconv_canal_up_no(is,isa)),0.0)
         avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
&         iconv_canal_up_no(is,isa))-struc_flow
       END IF
       IF (flnm(iflo).ne.name_struc_w_spec_code_wca(9)) THEN
         IF (n_add_grid_loc_s(is,isa).eq.0) pond(itwlc)=pond(itwlc)
&         +struc_flow*86400./(gdar*(1.0-carfact(itwlc)
$         -total_small_afact(itwlc)))
         DO i=1,n_add_grid_loc_s(is,isa)
           pond(node_add_s(is,isa,i))=pond(node_add_s(is,isa,i))+
&           struc_flow*86400./(n_add_grid_loc_s(is,isa)*gdar*
&           (1.0-carfact(node_add_s(is,isa,i))-total_small_afact
&           (node_add_s(is,isa,i))))
         END DO
       END IF
     END IF
     struc_flow=env_thru_struc(is,isa)+reg_thru_struc(is,isa)+
&     ws_thru_struc(is,isa)
     IF (flnm(iflo).ne.name_struc_w_spec_code_wca(9)) THEN
       IF (ihwopt(is,isa).eq.0) THEN
         qu(ihwlc)=qu(ihwlc)-struc_flow*86400.
         IF (ivary_slope_daily_opt(ihwlc).eq.1) THEN
           CALL match (cnm(ihwlc), 1, 5, canal_name_dvslope,
&           ncnl_dly_slope, 5, imatch)
           sgn=1
           IF (isuppflw_for_slope_wca(is,isa).eq.2) THEN
             sgn=-1
           END IF
           dischg_for_slope_kn(imatch,2,isuppflw_for_slope_wca(is,
&           isa))=dischg_for_slope_kn(imatch,2,
&           isuppflw_for_slope_wca(is,isa))+sgn*struc_flow
         END IF
       ELSE
```

```
          IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
&') THEN
            pond(ihwlc)=pond(ihwlc)-struc_flow*86400./(gdar*(1.0
&           -carfact(ihwlc)-total_small_afact(ihwlc)))
          END IF
        END IF
      ELSE
        IF (ica3.le.nch.and.itwopt(is,isa).eq.0) THEN
          CALL match ('S339  ', 1, 6, struc_name_meas, nflpts, 6,
&          imatch)
          IF (imatch.ne.0.or.(imatch.eq.0.and.runmode.eq.'SIMUL'))
&          THEN
            head_drop_s11=0.85*hdc(ica3,iseas)
          ELSE
            head_drop_s11=0.70*hdc(ica3,iseas)
          END IF
          IF (chdep(ica3)+head_drop_s11.lt.ells(nod_ds_s11ab)) THEN
            flow_from_s11_to_canal=struc_flow
          ELSE
            flow_from_s11_to_canal=0.0
          END IF
          avvolws(idn_wca_indx(is8bsn-6),iconvey_canal_no(is8bsn-6))
&          =avvolws(idn_wca_indx(is8bsn-6),iconvey_canal_no(is8bsn-
&          6))+flow_from_s11_to_canal
          avvolrg(idn_wca_indx(is8bsn-6),iconvey_canal_no(is8bsn-6))
&          =avvolrg(idn_wca_indx(is8bsn-6),iconvey_canal_no(is8bsn-
&          6))+flow_from_s11_to_canal
        END IF
      END IF
      rlocal_flow=kflo(iflo)
      kflo(iflo)=kflo(iflo)+ws_thru_struc(is,isa)+reg_thru_struc(is,
&      isa)+env_thru_struc(is,isa)
      IF (flnm(iflo).eq.name_struc_w_spec_code_wca(33).or.flnm(iflo)
&      .eq.name_struc_w_spec_code_wca(34)) THEN
        avvolws(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))=
&        avvolws(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))+
&        struc_flow
        avvolrg(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))=
&        avvolrg(idn_wca_indx(is6bsn-6),iconvey_canal_no(is6bsn-6))+
&        struc_flow
      END IF
      IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7)) wss39=
&      ws_thru_struc(is,isa)+wss39
      IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7)) regs39=
&      reg_thru_struc(is,isa)
      IF (flnm(iflo).eq.name_struc_w_spec_code_wca(7)) THEN
        add_demand_struc=add_demand_s39
        add_avail_vol_up=avail_vol_s39
      ELSE
        add_demand_struc=0.0
        add_avail_vol_up=0.0
      END IF
      IF (lp(23)) THEN
        WRITE (62,10) ifyr+iyear-1,month,iday,flnm(iflo),hw,tw,
&        struc_cap,avvol_upstream+add_avail_vol_up,struc_flow+
&        rlocal_flow,wsneed+add_demand_struc
      END IF
```

```
   10     FORMAT (i5,2i3,5x,a6,5x,2f7.2,8f10.0)
        ELSE
C
C  SPECIAL CODE UNIQUE FOR APPROPRIATE STRUCTURES
C
C-----------------------------------------------------------
C                   WATER SUPPLY FROM STA-1E TO C-51
C-----------------------------------------------------------
          IF (flnm(iflo).eq.name_struc_w_spec_code_wca(21)) THEN
C
C   deliver water from sta-1e first if water is available before
C    using wca-1
C
            IF (istaopt(5).eq.1) THEN
              avvol_sta1e=(hw-stgbpt_s(is,isa,1))*gdar/86400.
              avvol_sta1e=amax1(avvol_sta1e,0.0)

              IF (hw.gt.tw) THEN
                s319_sply_cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
              ELSE
                s319_sply_cap=0.0
              END IF
C
              wsdel_to_c51w=amin1(avvol_sta1e,s319_sply_cap,dqu(itwlc))
              kflo(iflo)=wsdel_to_c51w
              IF (itwopt(is,isa).eq.0) THEN
                dqu(canal_no_add_s(is,isa,1))=dqu(canal_no_add_s(is,isa,
     &           1))-wsdel_to_c51w
                dqu(canal_no_add_s(is,isa,1))=
     &           amax1(dqu(canal_no_add_s(is,isa,1)),0.0)
                dqu(itwlc)=dqu(itwlc)-wsdel_to_c51w
              END IF
C
              IF (iprorte(isa).eq.1) THEN
                tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &           iconv_canal_up_no(is,isa))-wsdel_to_c51w
              END IF
C
            END IF
C-------------------------------------------------------------
C               PROPOSED OUTLET FROM WCA2B
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(22)) THEN
C
            IF (sim_opt(is,isa).eq.'ENV') THEN
              IF (hw.gt.tw) THEN
                struc_cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
              ELSE
                struc_cap=0.0
              END IF
              itarg=0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  itarg=itarg+1
                END IF
              END DO
              iup_targ=iup_targ_indx_wca(is,isa)
```

```
              r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
              r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
              IF (iup_targ.gt.ntotal_no_import_areas) THEN
                r1top=999.
                r1bot=998.
              END IF
              IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                IF (itarg.eq.0) THEN
                  struc_outflow=2.0*struc_cap*
     &             (avg_sim_wca_stg_out(iup_targ)-r1top)
                  struc_outflow=amin1(struc_outflow,tvolwtr,struc_cap)
                ELSE
                  struc_outflow=struc_cap
                END IF
              ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
                struc_outflow=1/(r1top-r1bot)*struc_cap*
     &           (avg_sim_wca_stg_out(iup_targ)-r1bot)
                IF (itarg.eq.0) THEN
                  struc_outflow=amin1(struc_outflow,tvolwtr)
                END IF
              ELSE
                struc_outflow=0.0
              END IF
            ELSE
              struc_outflow=0.0
            END IF
c           print *,stage_outflw_wca(iup_targ),offset_out(iup_targ,1)
c    $      ,offset_out(iup_targ,2),r1top,r1bot,struc_outflow,' C2ALB1'
c    $      ,avg_sim_wca_stg_out(iup_targ)
            avail_water_to_pump=amax1(desgncp(is,isa)-
     &       total_outflow_env_2b,0.0)
            struc_outflow=amin1(avail_water_to_pump,struc_outflow)
            total_outflow_env_2b=total_outflow_env_2b+struc_outflow
            kflo(iflo)=struc_outflow
            IF (itarg.eq.0) tvolwtr=tvolwtr-kflo(iflo)
C-------------------------------------------------------------
C               PROPOSED OUTLET FROM WCA2B
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(23)) THEN
C
            IF (hw.gt.tw) THEN
              struc_cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              struc_cap=0.0
            END IF
            IF (sim_opt(is,isa).eq.'ENV') THEN
              itarg=0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  itarg=itarg+1
                END IF
              END DO
              iup_targ=iup_targ_indx_wca(is,isa)
              r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
              r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
              IF (iup_targ.gt.ntotal_no_import_areas) THEN
                r1top=999.
```

```
                r1bot=998.
              END IF
              IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                IF (itarg.eq.0) THEN
                  struc_outflow=2.0*struc_cap*
     &             (avg_sim_wca_stg_out(iup_targ)-r1top)
                  struc_outflow=amin1(struc_outflow,tvolwtr,struc_cap)
                ELSE
                  struc_outflow=struc_cap
                END IF
              ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
                struc_outflow=1/(r1top-r1bot)*struc_cap*
     &           (avg_sim_wca_stg_out(iup_targ)-r1bot)
                IF (itarg.eq.0) THEN
                  struc_outflow=amin1(struc_outflow,tvolwtr)
                END IF
              ELSE
                struc_outflow=0.0
              END IF
            ELSE
              struc_outflow=0.0
            END IF
            avail_water_to_pump=amax1(desgncp(is,isa)-
     &       total_outflow_env_2b,0.0)
            struc_outflow=amin1(avail_water_to_pump,struc_outflow)
            total_outflow_env_2b=total_outflow_env_2b+struc_outflow
            kflo(iflo)=struc_outflow
            IF (itarg.eq.0) tvolwtr=tvolwtr-kflo(iflo)

C-------------------------------------------------------------
C               PROPOSED OUTLET FROM WCA2B
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(24)) THEN
C
            IF (hw.gt.tw) THEN
              struc_cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              struc_cap=0.0
            END IF
            IF (sim_opt(is,isa).eq.'ENV') THEN
              itarg=0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  itarg=itarg+1
                END IF
              END DO
              iup_targ=iup_targ_indx_wca(is,isa)
              r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
              r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
              IF (iup_targ.gt.ntotal_no_import_areas) THEN
                r1top=999.
                r1bot=998.
              END IF
              IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                IF (itarg.eq.0) THEN
                  struc_outflow=2.0*struc_cap*
     &             (avg_sim_wca_stg_out(iup_targ)-r1top)
```

```
                  struc_outflow=amin1(struc_outflow,tvolwtr,struc_cap)
                ELSE
                  struc_outflow=struc_cap
                END IF
              ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
                struc_outflow=1/(r1top-r1bot)*struc_cap*
     &           (avg_sim_wca_stg_out(iup_targ)-r1bot)
                IF (itarg.eq.0) THEN
                  struc_outflow=amin1(struc_outflow,tvolwtr)
                END IF
              ELSE
                struc_outflow=0.0
              END IF
            ELSE
              struc_outflow=0.0
            END IF
            avail_water_to_pump=amax1(desgncp(is,isa)-
     &       total_outflow_env_2b,0.0)
            struc_outflow=amin1(avail_water_to_pump,struc_outflow)
            total_outflow_env_2b=total_outflow_env_2b+struc_outflow
            kflo(iflo)=struc_outflow
            IF (itarg.eq.0) tvolwtr=tvolwtr-kflo(iflo)
C-------------------------------------------------------------
C               PROPOSED OUTLET FROM WCA3A TO LAKE BELT
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(25)) THEN
C
            IF (hw.gt.tw) THEN
              struc_cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              struc_cap=0.0
            END IF
            av_vol=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*gdar
            avail_vol=amax1(av_vol,0.0)/86400.
            CALL match ('LBLTC', 1, 5, cnm, nch+1, 5, ilbtc)
            IF (ilbtc.eq.0) ilbtc=nch+1
C
            IF (sim_opt(is,isa).eq.'ENV') THEN
              itarg=0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  itarg=itarg+1
                END IF
              END DO
              iup_targ=iup_targ_indx_wca(is,isa)
              r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
              r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
              IF (iup_targ.gt.ntotal_no_import_areas) THEN
                r1top=999.
                r1bot=998.
              END IF
              IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                IF (itarg.eq.0) THEN
                  IF (ilkbeltc.le.ntotres) THEN
                    struc_outflow=2.0*struc_cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1top)
                    struc_outflow=amin1(struc_outflow,tvolwtr,struc_cap)
```

```
                  END IF
                  IF (ilbtc.le.nch) THEN
                    IF (chdep(ilbtc).ge.rmxstge(is,isa,1)) THEN
                      struc_outflow=0.0
                    ELSE IF (chdep(ilbtc).gt.rmxstge(is,isa,1)-
     &               0.2.and.chdep(ilbtc).lt.rmxstge(is,isa,1)) THEN
                      struc_outflow=5.0*struc_outflow*(rmxstge(is,isa,1)
     &                 -chdep(ilbtc))
                    END IF
                  END IF
                ELSE
                  struc_outflow=struc_cap
                END IF
              ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
                struc_outflow=1/(r1top-r1bot)*struc_cap*
     &           (avg_sim_wca_stg_out(iup_targ)-r1bot)
                IF (itarg.eq.0.and.ilkbeltc.le.ntotres) THEN
                  struc_outflow=amin1(struc_outflow,tvolwtr)
                END IF
              ELSE
                struc_outflow=0.0
              END IF
            ELSE
              struc_outflow=0.0
            END IF
            IF (itarg.gt.0.or.ilkbeltc.gt.ntotres) THEN
              avail_water_to_pump=amax1(desgncp(is,isa)-
     &         total_outflow_env_2b,0.0)
            ELSE
              avail_water_to_pump=99999.
            END IF
            struc_outflow=amin1(avail_water_to_pump,struc_outflow,
     &       avail_vol)
            kflo(iflo)=struc_outflow
            IF (itarg.eq.0) tvolwtr=tvolwtr-kflo(iflo)
C
C-----------------------------------------------------------------
C                   WATER SUPPLY TO L-8 FROM WCA-1
C                      THRU S-5AS
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(3)) THEN
C
            IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL  ')
     &        THEN
              IF (istaopt(is5absn-6).eq.0) THEN
                add_inflow=tinflow(isa,iconv_canal_up_no(is,isa))
              ELSE
                add_inflow=0.0
              END IF
C
C       DELIVER W.S. TO L-8 FROM WCA-1
C
              IF (ihwopt(is,isa).eq.0) THEN
                avvoltol8=(chdep(ihwlc)-chdep(itwlc))*cl(ihwlc)*
     &           width(ihwlc)+cgseep(ihwlc,chdep(itwlc),0,0.)+
     &           tovflc(ihwlc,chdep(itwlc),0,0.,0,1)+add_inflow*86400.+
```

```
     &           addflw(isa,iconv_canal_up_no(is,isa))*86400.
               ELSE
                 avvoltol8=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*
     &           gdar
               END IF
             ELSE
               avvoltol8=avail_vol_pipel(iserv_area_indx(is,isa))*86400.
             END IF
C
             IF (opt_l8_prop) THEN
               IF (itwopt(is,isa).eq.0) THEN
                 wsneed_l8=dqu(itwlc)
               ELSE
                 wsneed_l8=wsneedl8
               END IF
             ELSE
               wsneed_l8=wsneedl8
             END IF
C
             avvoltol8=amax1(avvoltol8,0.0)/86400.
             IF (nint(wsneed_l8).gt.0) THEN
               IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &') THEN
                 IF (ihwopt(is,isa).eq.0) THEN
                   IF (iprorte(isa).eq.1) THEN
                     ratio=avvolws(isa,iconv_canal_up_no(is,isa))/
     &                tneed(isa,iconv_canal_up_no(is,isa))
                     ratio=amin1(ratio,1.0)
                     wsdel=wsneed_l8*ratio
                     cout=amin1(wsdel,avvoltol8)*wsctb*
     &                frac_del_frm_wca(isa)
                     tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &                iconv_canal_up_no(is,isa))-wsneed_l8
                   ELSE
                     cout=amin1(wsneed_l8,avvoltol8,avvolws(isa,
     &                iconv_canal_up_no(is,isa)))*wsctb*
     &                frac_del_frm_wca(isa)
                   END IF
                   ratiows=cout/wsneed_l8
                 ELSE
                   cout=amin1(avvoltol8,wsneed_l8)*frac_del_frm_wca(isa)
                   ratiows=cout/wsneed_l8
                 END IF
               ELSE
                 ratiows=avvoltol8/tneed_pipel(iserv_area_indx(is,isa))
                 ratiows=amin1(ratiows,1.0)
                 cout=ratiows*wsneed_l8*frac_del_frm_wca(isa)
                 tneed_pipel(iserv_area_indx(is,isa))=
     &            tneed_pipel(iserv_area_indx(is,isa))-wsneed_l8
                 avail_vol_pipel(iserv_area_indx(is,isa))=
     &            avail_vol_pipel(iserv_area_indx(is,isa))-cout
               END IF
             ELSE
               cout=0.0
               ratiows=0.0
             END IF
C
```

```
            IF (opt_l8_prop) THEN
              pond(node_add_s(is,isa,1))=pond(node_add_s(is,isa,1))+
     &         cout*86400./(gdar*(1.0-carfact(node_add_s(is,isa,1))
     &         - total_small_afact(node_add_s(is,isa,1))))
              IF (ihwopt(is,isa).eq.0) THEN
                qu(ihwlc)=qu(ihwlc)-cout*86400.
              ELSE
                IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &  ') THEN
                  pond(ihwlc)=pond(ihwlc)-cout*86400./(gdar*(1.0
     &            -carfact(ihwlc)-total_small_afact(ihwlc)))
                END IF
              END IF
              IF (itwopt(is,isa).eq.0) THEN
                dqu(itwlc)=dqu(itwlc)-cout
              END IF
              wsdel_to_l8s=cout
            ELSE
              IF (istaopt(is5absn-6).eq.1) THEN
                pond(node_add_s(is,isa,1))=pond(node_add_s(is,isa,1))+
     &           cout*86400./(gdar*(1.0-carfact(node_add_s(is,isa,1))
     &           -total_small_afact(node_add_s(is,isa,1))))
                IF (ihwopt(is,isa).eq.0) THEN
                  qu(ihwlc)=qu(ihwlc)-cout*86400.
                ELSE
                  IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIP
     &EL  ') THEN
                    pond(ihwlc)=pond(ihwlc)-cout*86400./(gdar*(1.0
     &              -carfact(ihwlc)-total_small_afact(ihwlc)))
                  END IF
                END IF
              END IF
              IF (itwopt(is,isa).eq.0) THEN
                qu(itwlc)=qu(itwlc)+ratiows*min(iagflo(iag(il8agbsn),1),
     &           0)*86400.
              ELSE
                pond(itwlc)=pond(itwlc)+ratiows*min(iagflo(iag(il8agbsn)
     &           ,1),0)*86400./(gdar*(1.0-carfact(itwlc)
     &           -total_small_afact(itwlc)))
              END IF
              IF (iagflo(iag(il8agbsn),1).lt.0) THEN
                iagflo(iag(il8agbsn),1)=iagflo(iag(il8agbsn),1)*(1-
     &           ratiows)
              END IF
              wsneedl8=wsneedl8-cout
              wsdel_to_l8=cout
            END IF
            kflo(add_dep_str_indx_s(is,isa,2))=
     &       kflo(add_dep_str_indx_s(is,isa,2))-cout
            kflo(iflo)=kflo(iflo)-cout
            s5a2_no=s5a2_no+cout
            s5a3_no=s5a3_no+cout
C
C  deliver regulatory releases from WCA-1 to Tide via C-51
C
            IF (iopt_for_lokreg_flwth(is,isa).eq.0) THEN
              IF (iopt_for_reg_releases(is,isa)
```

```
&          .eq.1.and.trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
&          .and.sim_opt(is,isa).eq.'REG'.and..not.opt_l8_prop) THEN
            IF (rmulti_seas_pred(month)
&            .gt.rmulti_seas_thres_struc(is,isa)
&            .or.iatl_convyr_index.eq.1) THEN
             avail_cap_reg=amax1(s5a4max-flfrml8toc51w,0.0)
             reg_del_to_c51=amin1(avail_cap_reg,avvolrg(isa,
&             iconv_canal_up_no(is,isa)))
             cout=cout+reg_del_to_c51
             IF (istaopt(is5absn-6).eq.0) THEN
               kflo(iflo)=kflo(iflo)-reg_del_to_c51
             ELSE
               IF (ihwopt(is,isa).eq.0) THEN
                 qu(ihwlc)=qu(ihwlc)-reg_del_to_c51*86400.
               ELSE
                 IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa))
&                 .ne.'PIPEL  ') THEN
                  pond(ihwlc)=pond(ihwlc)-reg_del_to_c51*86400./
&                  (gdar*(1.0-carfact(ihwlc)-total_small_afact
&                  (ihwlc)))
                 END IF
               END IF
               CALL match ('WPCB ', 1, 5, cnm, nch, 5, iwpcbc)
               qu(iwpcbc)=qu(iwpcbc)+reg_del_to_c51*86400.
             END IF
             s5a2_no=s5a2_no+reg_del_to_c51
            ELSE
             reg_del_to_c51=0.0
            END IF
          ELSE
            reg_del_to_c51=0.0
          END IF
        ELSE
          IF (iopt_for_reg_releases(is,isa).eq.1.and.sim_opt(is,isa)
&          .eq.'REG') THEN
            avail_cap_reg=amax1(s5a4max-flfrml8toc51w,0.0)
            IF (trig_wca_stg(isa).le.regstg_wca(isa,iwzone))
&            avail_cap_reg=amin1(avail_cap_reg,
&            reg_rel_frm_lok(ieaa_conduit_indx(is,isa)),avvolws(isa,
&            iconv_canal_up_no(is,isa)))
            reg_del_to_c51=avail_cap_reg
            cout=cout+reg_del_to_c51
            kflo(iflo)=kflo(iflo)-reg_del_to_c51
            s5a2_no=s5a2_no+reg_del_to_c51
          ELSE
            reg_del_to_c51=0.0
          END IF
        END IF
        IF (ihwopt(is,isa).eq.0) THEN
          avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
&          iconv_canal_up_no(is,isa))-cout
          avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
&          iconv_canal_up_no(is,isa))-cout
          avvolrg(isa,iconv_canal_up_no(is,isa))=amax1(avvolrg(isa,
&          iconv_canal_up_no(is,isa)),0.0)
        END IF
C
```

```
C     deliver water from L-8 to C-51
C
            CALL match (name_res_spec_code(21), 1, 6, resname, ntotres,
     &       6, ista1e_res)
            IF (ista1e_res.eq.0) THEN
              flws319=0.0
            END IF
            IF (chdep(itwlc).gt.chdep(canal_no_add_s(is,isa,1))+
     &       hdc(canal_no_add_s(is,isa,1),iseas).and.nint(flws319)
     &       .eq.0.and.kflo(add_dep_str_indx_s(is,isa,3)).ge.0.0) THEN
              capac=s5a4max-kflo(add_dep_str_indx_s(is,isa,3))
              IF (dqu(canal_no_add_s(is,isa,1)).gt.0.0) THEN
                cout=amin1(dqu(canal_no_add_s(is,isa,1)),capac)
              ELSE
                cout=0.0
              END IF
              kflo(add_dep_str_indx_s(is,isa,2))=
     &         kflo(add_dep_str_indx_s(is,isa,2))+cout
              dqu(canal_no_add_s(is,isa,1))=dqu(canal_no_add_s(is,isa,1)
     &         )-cout
              s5a3_so=s5a3_so+cout
            ELSE IF (nint(flws319).eq.0) THEN
              volout=0.0
              tot_pump=0.0
              DO ns=1,nstrctr(canal_no_add_s(is,isa,1))
                volout=strout(stgbpt_s(is,isa,1),canal_no_add_s(is,isa,
     &           1),canal_no_add_s(is,isa,1),ns,0,crmin,volwtr,prevqu,
     &           rain,0,tot_pump,jday,tempf_volwtr,tempf_weircff,0)+
     &           volout
              END DO
              volout=volout*86400.
              avvolb=(chdep(canal_no_add_s(is,isa,1))-stgbpt_s(is,isa,1)
     &         +hdc(canal_no_add_s(is,isa,1),iseas))*
     &         cl(canal_no_add_s(is,isa,1))*width(canal_no_add_s(is,isa,
     &         1))+cgseep(canal_no_add_s(is,isa,1),stgbpt_s(is,isa,1)-
     &         hdc(canal_no_add_s(is,isa,1),iseas),0,0.)+
     &         tovflc(canal_no_add_s(is,isa,1),stgbpt_s(is,isa,1)-
     &         hdc(canal_no_add_s(is,isa,1),iseas),0,0.,0,1)+drainc51w-
     &         volout
              avvolb=amax1(avvolb/86400.,0.0)
              avals5a=s5amax-kflo(add_dep_str_indx_s(is,isa,4))
              IF (chdep(canal_no_add_s(is,isa,1))+hdc(canal_no_add_s(is,
     &         isa,1),iseas).gt.tw) THEN
                IF (chdep(canal_no_add_s(is,isa,1))+
     &           hdc(canal_no_add_s(is,isa,1),iseas).gt.tw+0.25) THEN
                  capac_w_s5a4=s5a4max
                ELSE
                  capac_w_s5a4=s5a4max*4.0*(chdep(canal_no_add_s(is,isa,
     &             1))+hdc(canal_no_add_s(is,isa,1),iseas)-tw)
                END IF
                cout=-amin1(avvolb,capac_w_s5a4,avals5a)
              ELSE
                cout=0.0
              END IF
              IF (istaopt(is5absn-6).eq.0.or.opt_l8_prop) THEN
                kflo(iflo)=kflo(iflo)-cout
              ELSE
```

```
                CALL match ('WPCB ', 1, 5, cnm, nch, 5, iwcpb)
                qu(iwcpb)=qu(iwcpb)-abs(cout)*86400.
                qu(ihwlc)=qu(ihwlc)+abs(cout)*86400.
              END IF
              s5a2_so=s5a2_so+abs(cout)
            ELSE
              cout=0.0
            END IF
            dqu(canal_no_add_s(is,isa,1))=dqu(canal_no_add_s(is,isa,1))-
     &       amin1(reg_del_to_c51,dqu(canal_no_add_s(is,isa,1)))
            kflo(add_dep_str_indx_s(is,isa,3))=
     &       kflo(add_dep_str_indx_s(is,isa,3))+cout+reg_del_to_c51
C-----------------------------------------------------------------
C                     G-94C INTO LAKE WORTH
C                     DRAINAGE DISTRICT FROM WCA1
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq. 'G94C  ') THEN
            IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL  ')
     &        THEN
              IF (hw .gt. tw) THEN
                qmax = dcoeff(is,isa)*(hw-tw)**power(is,isa)
              ELSE
                qmax = 0.0
              ENDIF
              IF (ihwopt(is,isa).eq.0) THEN
                IF (runmode.eq.'SIMUL') THEN
                  IF (rlok_to_sa1.gt.0.0) THEN
                    avail_vol_wca1=avvol_rlok_to_sa1
                  ELSE
                    avail_vol_wca1=avvolws(isa,iconv_canal_up_no(is,isa)
     &               )
                  END IF
                ELSE
                  avail_vol_wca1=avvolws(isa,iconv_canal_up_no(is,isa))
                END IF
                twsneed_rem=wsnde1+wsnde2+wsnde3+wsndese
                IF (nint(twsneed).gt.0.and.itwopt(is,isa).eq.0) THEN
                  wsfact=wsctb
                  IF (iprorte(isa).eq.1) THEN
                    ratio=avail_vol_wca1/tneed(isa,iconv_canal_up_no(is,
     &               isa))
                    avvol_upstream=ratio*twsneed - total_outf_g94s
                    ratio=amin1(ratio,1.0)
                    wsdel=ratio*twsneed - total_outf_g94s
                    cout=amin1(qmax,wsdel)*wsctb*frac_del_frm_wca(isa)
                    print *,ratio,avail_vol_wca1,avvol_rlok_to_sa1
     &               ,tneed(isa,iconv_canal_up_no(is,isa))
     &               ,rlok_to_sa1,'  G94C ',cout
                  ELSE
                    ratio=1.0
                    wsdel=twsneed_rem
                    cout=amin1(wsdel,qmax,avail_vol_wca1)*wsctb*
     &               frac_del_frm_wca(isa)
                    avvol_upstream=avail_vol_wca1
                  END IF
                  IF (nint(wsdel).gt.0) THEN
                    ratcapc=cout/(wsdel*wsctb)
```

```
                ELSE
                  ratcapc=0.0
                END IF
              ELSE
                cout=0.0
                wsfact=1.0
                ratcapc=0.0
                ratio=1.0
                avvol_upstream=avail_vol_wca1
              END IF
C
              avvolws_orig=avvolws(isa,iconv_canal_up_no(is,isa))
              avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
     &         iconv_canal_up_no(is,isa))-cout
              avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &         iconv_canal_up_no(is,isa))-cout
            ELSE
              twsneed=wsnde1+wsnde2+wsnde3+wsndese
              avail_vol_ws=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*
     &         gdar/86400.
              avail_vol_ws=amax1(avail_vol_ws,0.0)
              avvol_upstream=avail_vol_ws
              cout=amin1(twsneed,qmax,avail_vol_ws)*
     &         frac_del_frm_wca(isa)
              ratio=1.0
              wsfact=1.0
              IF (twsneed.gt.0.0) THEN
                ratcapc=cout/twsneed
              ELSE
                ratcapc=0.0
              END IF
            END IF
          ELSE
            wsfact=1.0
            IF (nint(twsneed).gt.0) THEN
              ratio=avail_vol_pipel(iserv_area_indx(is,isa))/
     &         tneed_pipel(iserv_area_indx(is,isa))
              avvol_upstream=ratio*twsneed-total_outf_g94s
              ratio = min(ratio,1.0)
              wsdel=ratio*twsneed - total_outf_g94s
              cout=wsdel*frac_del_frm_wca(isa)
              IF (nint(wsdel).gt.0) THEN
                ratcapc=cout/wsdel
              ELSE
                ratcapc=0.0
              END IF
            ELSE
              ratcapc=0.0
              avvol_upstream=0.0
              ratio=1.0
              cout=0.0
            END IF
          END IF
          kflo(iflo)=cout
          total_outf_g94s = total_outf_g94s + cout
          IF (lp(23)) THEN
            WRITE (62,10) ifyr+iyear-1,month,iday,flnm(iflo),hw,
```

```
     &         tw,qmax,avvol_upstream,cout,twsneed
            END IF
C
            IF (ihwopt(is,isa).eq.0) THEN
              qu(ihwlc)=qu(ihwlc)-cout*86400.
            ELSE
              IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &        ') THEN
                pond(ihwlc)=pond(ihwlc)-cout*86400./(gdar*(1.0-
     &          carfact(ihwlc)-total_small_afact(ihwlc)))
              END IF
            END IF
            wsfact=wsfact*ratcapc*ratio
            qu(ie1)=qu(ie1)+wsnde1*wsfact*86400.
            qu(ie2)=qu(ie2)+wsnde2*wsfact*86400.
            qu(ie3)=qu(ie3)+wsnde3*wsfact*86400.
            qu(iese)=qu(iese)+wsndese*wsfact*86400.
            wsndese=(1-wsfact)*wsndese
            wsnde1=(1-wsfact)*wsnde1
            wsnde2=(1-wsfact)*wsnde2
            wsnde3=(1-wsfact)*wsnde3
C-----------------------------------------------------------------
C
C-----------------------------------------------------------------
C                     G-94AB (GATED CULVERTS)
C            WITH S-4 AND S-3 PUMPAGE INTO LAKE WORTH
C              DRAINAGE DISTRICT CANALS FROM WCA1
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(1)) THEN
            IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL  ')
     &        THEN
              IF (hw.gt.stgbpt_s(is,isa,1)) THEN
                qmax=dcoeff(is,isa)
              ELSE IF (hw.gt.stgbpt_s(is,isa,2)) THEN
                qmax=dcoeff(is,isa)/(stgbpt_s(is,isa,1)-stgbpt_s(is,isa,
     &           2))*(hw-stgbpt_s(is,isa,2))
              ELSE
                qmax=0.0
              END IF
              IF (ihwopt(is,isa).eq.0) THEN
                IF (runmode.eq.'SIMUL') THEN
                  IF (rlok_to_sa1.gt.0.0) THEN
                    avail_vol_wca1=avvol_rlok_to_sa1
                  ELSE
                    avail_vol_wca1=avvolws_orig
                  END IF
                ELSE
                  avail_vol_wca1=avvolws(isa,iconv_canal_up_no(is,isa))
                END IF
C
                twsneed_rem=wsnde1+wsnde2+wsnde3+wsndese
                IF (nint(twsneed).gt.0.and.itwopt(is,isa).eq.0) THEN
                  wsfact=wsctb
                  IF (iprorte(isa).eq.1) THEN
                    ratio=avail_vol_wca1/tneed(isa,iconv_canal_up_no(is,
     &               isa))
```

```
                  avvol_upstream=ratio*twsneed - total_outf_g94s
                  ratio=amin1(ratio,1.0)
                  wsdel=ratio*twsneed - total_outf_g94s
                  cout=amin1(qmax,wsdel)*wsctb*frac_del_frm_wca(isa)
                  print *,avail_vol_wca1,tneed(isa,iconv_canal_up_no
     &             (is,isa)),ratio,qmax,wsdel,avvolws(isa,
     &             iconv_canal_up_no(is,isa)),'  G94AB  '
     &             ,total_outf_g94s,twsneed,month,iday
                ELSE
                  ratio=1.0
                  wsdel=twsneed_rem
                  cout=amin1(wsdel,qmax,avail_vol_wca1)*wsctb*
     &             frac_del_frm_wca(isa)
                  avvol_upstream=avail_vol_wca1
                END IF
                IF (nint(wsdel).gt.0) THEN
                  ratcapc=cout/(wsdel*wsctb)
                ELSE
                  ratcapc=0.0
                END IF
              ELSE
                cout=0.0
                wsfact=1.0
                ratcapc=0.0
                ratio=1.0
                avvol_upstream=avail_vol_wca1
              END IF
C
              tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &        iconv_canal_up_no(is,isa))-twsneed_lwdd
              avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
     &         iconv_canal_up_no(is,isa))-cout
              avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &         iconv_canal_up_no(is,isa))-cout
            ELSE
              twsneed=wsnde1+wsnde2+wsnde3+wsndese
              avail_vol_ws=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*
     &         gdar/86400.
              avail_vol_ws=amax1(avail_vol_ws,0.0)
              avvol_upstream=avail_vol_ws
              cout=amin1(twsneed,qmax,avail_vol_ws)*
     &         frac_del_frm_wca(isa)
              ratio=1.0
              wsfact=1.0
              IF (twsneed.gt.0.0) THEN
                ratcapc=cout/twsneed
              ELSE
                ratcapc=0.0
              END IF
            END IF
          ELSE
            wsfact=1.0
            twsneed=wsnde1+wsnde2+wsnde3+wsndese
            IF (nint(twsneed).gt.0) THEN
              ratio=avail_vol_pipel(iserv_area_indx(is,isa))/
     &         tneed_pipel(iserv_area_indx(is,isa))
              wsdel=ratio*twsneed - total_outf_g94s
```

```
                cout=wsdel*frac_del_frm_wca(isa)
                IF (nint(wsdel).gt.0) THEN
                  ratcapc=cout/wsdel
                ELSE
                  ratcapc=0.0
                END IF
                avvol_upstream=ratio*twsneed-total_outf_g94s
              ELSE
                ratcapc=0.0
                avvol_upstream=0.0
                ratio=1.0
                cout=0.0
              END IF
              tneed_pipel(iserv_area_indx(is,isa))=
     &         tneed_pipel(iserv_area_indx(is,isa))-twsneed_lwdd
              avail_vol_pipel(iserv_area_indx(is,isa))=
     &         avail_vol_pipel(iserv_area_indx(is,isa))-cout
     &         -total_outf_g94s
            END IF
            kflo(iflo)=cout
            total_outf_g94s = total_outf_g94s + cout
            IF (lp(23)) THEN
              WRITE (62,10) ifyr+iyear-1,month,iday,flnm(iflo),hw,
     &         stgbpt_s(is,isa,1),qmax,avvol_upstream,cout,twsneed
            END IF
C
            wsfact=wsfact*ratcapc*ratio
C
C   if supply to lwdd is limited by capacity,then route water from wca-1
C   to c-51 and lwdd
C
            IF (runmode.eq.'SIMUL') THEN
C
C  remaining water supply needs of lwdd canals
C
              rem_wsndese=(1-wsfact)*wsndese
              rem_wsndes=wsndes
              rem_wsnde1=(1-wsfact)*wsnde1
              rem_wsnde2=(1-wsfact)*wsnde2
              rem_wsnde3=(1-wsfact)*wsnde3
              total_rem_dmnd_lwdd=rem_wsndese+rem_wsnde1+
     &         rem_wsnde2+rem_wsnde3
              IF (ihwopt(is,isa).eq.0) THEN
                iconvey_c=iconv_canal_up_no(is,isa)
                IF (iprorte(isa).eq.1) THEN
                  IF (nint(total_rem_dmnd_lwdd).gt.0) THEN
                    ratio_rem_deliv=((avvolws(isa,iconvey_c)+
     &               total_outf_g94s)*twsneed/tneed_orig(isa,iconvey_c)
     &               -total_outf_g94s)/total_rem_dmnd_lwdd
                    ratio_rem_deliv=amin1(ratio_rem_deliv,1.0)
                  ELSE
                    ratio_rem_deliv=1.0
                  END IF
                ELSE
                  ratio_rem_deliv=1.0
                END IF
              ELSE
```

```
                IF (istaopt(is5absn-6).eq.0) THEN
                  ihwlc_c51ws=30-minx(53)+1+isum(53)
                ELSE
                  ihwlc_c51ws=30-minx(52)+1+isum(52)
                END IF
                dpthgw=(ells(ihwlc_c51ws)-h(ihwlc_c51ws))*s(ihwlc_c51ws)
                IF (pond(ihwlc_c51ws).gt.dpthgw) THEN
                  hw_c51ws=pond(ihwlc_c51ws)-dpthgw+ells(ihwlc_c51ws)
                ELSE
                  hw_c51ws=(pond(ihwlc_c51ws)-dpthgw)/s(ihwlc_c51ws)+
     &             ells(ihwlc_c51ws)
                END IF

                avail_vol_ws=(hw_c51ws-ells(ihwlc_c51ws)-
     &           deten_def(lutyp(ihwlc_c51ws)))*gdar/86400.
                IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &  ') THEN
                  ratio_rem_deliv=avail_vol_ws/total_rem_dmnd_lwdd
                ELSE
                  ratio_rem_deliv=0.0
                END IF
              END IF
C
              IF (opt_l8_prop) THEN
                itw=il8s
              ELSE
                itw=il8
              END IF
              total_add_supply_to_lwdd=0.0
C
              IF (ihwopt(is,isa).ne.0) THEN
                hw_op=hw_c51ws
              ELSE
                hw_op=hw
              END IF
              IF (hw_op.gt.chdep(itw)) THEN
                avvol_to_c51=amin1(ratio_rem_deliv*total_rem_dmnd_lwdd,
     &           s5a4max-kflo(iflbel5))
                IF (iprorte(isa).ne.1.and.ihwopt(is,isa).eq.0) THEN
                  avvol_to_c51=amin1(avvol_to_c51,avvolws(isa,iconvey_c)
     &             )
                END IF
                avvol_to_c51=amax1(avvol_to_c51,0.0)
C             print *,avvol_to_c51,ratio_rem_deliv,total_rem_dmnd_lwdd
C    $            ,avvolws(isa,iconvey_c),'   BBBB'
C
C  remaining capacities of pumps into lwdd from c-51
C
                rem_capac_cs9=capaccs9-kflo(iflcs9)
                rem_capac_cs12=capaccs12-kflo(iflcs12)
                rem_capac_cs2=capaccs2-kflo(iflcs2)
                rem_capac_cs4=capaccs4-kflo(iflcs4)
C
                total_capac_east=rem_capac_cs9+rem_capac_cs12
                add_supply_to_ese=amin1(total_capac_east,avvol_to_c51,
     &           rem_wsndese)*wsctb*frac_del_frm_wca(isa)
C             print *,add_supply_to_ese,total_capac_east
```

```
C    $        ,avvol_to_c51,rem_wsndese,'  C51TOESE'
               qu(iese)=qu(iese)+add_supply_to_ese*86400.
               avvol_to_c51=avvol_to_c51-add_supply_to_ese
               IF (ihwopt(is,isa).eq.0) THEN
                 avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
     &            add_supply_to_ese
                 avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
     &            add_supply_to_ese
               END IF
               rem_wsndese=rem_wsndese-add_supply_to_ese
               total_capac_east=total_capac_east-add_supply_to_ese
C
c              add_supply_to_es=amin1(total_capac_east,avvol_to_c51,
c    &          rem_wsndes)*wsctb*frac_del_frm_wca(isa)
c              avvol_to_c51=avvol_to_c51-add_supply_to_es
c              IF (ihwopt(is,isa).eq.0) THEN
c                avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
c    &            add_supply_to_es
c                avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
c    &            add_supply_to_es
c              END IF
c              qu(ies)=qu(ies)+add_supply_to_es*86400.
               qu(ie2)=qu(ie2)-add_supply_to_ese*
     &          86400.
c              rem_wsndes=rem_wsndes-add_supply_to_es
C
C  remaining avail water after deliv to lwdd canals near Boca wellfield
C
               rem_capac_east=total_capac_east
C
C  supply to E-2
C
               tot_supply_to_e2e3=rem_wsnde2+rem_wsnde3
               add_supply_to_e2=amin1(rem_capac_east,
     &          tot_supply_to_e2e3,avvol_to_c51)*wsctb*
     &          frac_del_frm_wca(isa)
C            print *,add_supply_to_e2,rem_capac_east
C    $        ,avvol_to_c51,tot_supply_to_e2e3,'  C51TO E2E3'
               IF (tot_supply_to_e2e3.gt.0.0) THEN
                 ratio_e2e3=add_supply_to_e2/tot_supply_to_e2e3
               ELSE
                 ratio_e2e3=0.0
               END IF
               total_rem_capac=rem_capac_east-add_supply_to_e2
               avvol_to_c51=avvol_to_c51-add_supply_to_e2
               IF (ihwopt(is,isa).eq.0) THEN
                 avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
     &            add_supply_to_e2
                 avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
     &            add_supply_to_e2
               END IF
               qu(ie3)=qu(ie3)+ratio_e2e3*rem_wsnde3*86400.
               qu(ie2)=qu(ie2)-ratio_e2e3*rem_wsnde3*86400.
               rem_wsnde2=rem_wsnde2*(1-ratio_e2e3)
               rem_wsnde3=rem_wsnde3*(1-ratio_e2e3)
C
C  supply to E-1
```

```
C
                add_supply_to_e1=amin1(rem_capac_cs2,rem_wsnde1,
     &           avvol_to_c51)*wsctb*frac_del_frm_wca(isa)
C             print *,add_supply_to_e1,rem_capac_cs2,rem_wsnde1
C    $         ,avvol_to_c51,wsctb,'  C51TO E1'
                IF (ihwopt(is,isa).eq.0) THEN
                  avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
     &             add_supply_to_e1
                  avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
     &             add_supply_to_e1
                END IF
                rem_wsnde1=rem_wsnde1-add_supply_to_e1
C
C  total supply deliv to lwdd via C-51
C
                total_add_supply_to_lwdd=add_supply_to_e2+
     &           add_supply_to_e1+add_supply_to_ese
C
                kflo(iflbel5)=kflo(iflbel5)+total_add_supply_to_lwdd
C
                IF (istaopt(is5absn-6).eq.1) THEN
                  pond(nodl101)=pond(nodl101)+total_add_supply_to_lwdd*
     &             86400./gdar
                  IF (ihwopt(is,isa).eq.0) THEN
                    qu(ihwlc)=qu(ihwlc)-total_add_supply_to_lwdd*86400.
                  ELSE
                    pond(ihwlc)=pond(ihwlc)-total_add_supply_to_lwdd*
     &               86400./(gdar*(1.0-carfact(ihwlc)-total_small_afact
     &               (ihwlc)))
                  END IF
                END IF
C
                kflo(ifls5a5)=kflo(ifls5a5)-total_add_supply_to_lwdd
                s5a2_no=s5a2_no+total_add_supply_to_lwdd
                ig124s = 0
                do ist = 1,nstrctr(ic51w)
                   if (tmpname(ic51w,ist) .eq. 'G124 ') ig124s = 1
                enddo
                if (ig124s.eq.1) kflo(iflg124)=total_add_supply_to_lwdd
                if (ig124s.eq.1) then
                  qu(ic51w)=qu(ic51w)-total_add_supply_to_lwdd*86400.
                  qu(ic51)=qu(ic51)+total_add_supply_to_lwdd*86400.
                else
                  out_flow_east0 = total_add_supply_to_lwdd
c                 print *,out_flow_east0,total_add_supply_to_lwdd,ig124s
c    &             ,'  RSDFTY'
                endif
                kflo(iflcs9)=capaccs9-capaccs9/(capaccs9+capaccs12)*
     &           total_rem_capac
                kflo(iflcs12)=capaccs12-capaccs12/(capaccs9+capaccs12)*
     &           total_rem_capac
                kflo(iflcs2)=kflo(iflcs2)+add_supply_to_e1
                kflo(iflcs4)=kflo(iflcs4)+add_supply_to_e3
              END IF
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs2
```

```
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs4
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs9
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs12
C
C  additional water delivered via Hillsboro Canal,if necessary
C
              total_rem_dmnd_after_c51=rem_wsndes+
     &         rem_wsnde2+rem_wsnde3+rem_wsnde1
C
              IF (ihwopt(is,isa).eq.0) THEN
                avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)+
     &           outflow_from_sta2_for_ws
              END IF
              IF (nint(twsneed).gt.0) THEN
                IF (ihwopt(is,isa).eq.0) THEN
                  IF (iprorte(isa).eq.1) THEN
                    avail_vol_s39=avvolws_orig*(twsneed/tneed_orig(isa,
     &              iconvey_c))-total_outf_g94s-total_add_supply_to_lwdd
     &              +outflow_from_sta2_for_ws
                  ELSE
                    avail_vol_s39=avvolws(isa,iconvey_c)
                  END IF
C             print *,avail_vol_s39,avvolws_orig,twsneed
C    $          ,cout,total_add_supply_to_lwdd
C    $          ,tneed_orig(isa,iconvey_c),'  AVVOLS39LWDD'
                ELSE
                  IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIP
     &EL  ') THEN
                    ihwlc_wss39=32-minx(43)+1+isum(43)
                    dpthgw=(ells(ihwlc_wss39)-h(ihwlc_wss39))*
     &               s(ihwlc_wss39)
                    IF (pond(ihwlc_wss39).gt.dpthgw) THEN
                      hw_wss39=pond(ihwlc_wss39)-dpthgw+
     &                 ells(ihwlc_wss39)
                    ELSE
                      hw_wss39=(pond(ihwlc_wss39)-dpthgw)/s(ihwlc_wss39)
     &                 +ells(ihwlc_wss39)
                    END IF
                    avail_vol_s39=(hw_wss39-ells(ihwlc_wss39)-
     &               deten_def(lutyp(ihwlc_wss39)))*gdar/86400.
                  ELSE
                    avail_vol_s39=0.0
                  END IF
                END IF
                avail_vol_s39=amax1(avail_vol_s39,0.0)
                rem_capac_cs17e=capaccs17e-kflo(iflcs17e)
                rem_capac_cs17w=capaccs17w-kflo(iflcs17w)
                avvol_to_hill=amin1(avail_vol_s39+regs39,s39cap-wss39)
c               add_supply_to_ese=0.0
c               add_supply_to_ese=amin1(rem_capac_cs17e,avvol_to_hill,
c    &           rem_wsndese)*frac_del_frm_wca(isa)
```

```
c                 IF (ihwopt(is,isa).eq.0) THEN
c                   avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
c     &              add_supply_to_ese
c                   avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
c     &              add_supply_to_ese
c                 END IF
c                 avvol_to_hill=avvol_to_hill-add_supply_to_ese
c                 rem_capac_cs17e=rem_capac_cs17e-add_supply_to_ese
c                 qu(iese)=qu(iese)+add_supply_to_ese*86400.
C
c                 add_supply_to_es=amin1(rem_capac_cs17e,avvol_to_hill,
c     &            rem_wsndes)*frac_del_frm_wca(isa)
c                 IF (ihwopt(is,isa).eq.0) THEN
c                   avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
c     &              add_supply_to_es
c                   avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
c     &              add_supply_to_es
c                 END IF
c                 avvol_to_hill=avvol_to_hill-add_supply_to_es
c                 rem_capac_cs17e=rem_capac_cs17e-add_supply_to_es
c                 qu(ies)=qu(ies)+add_supply_to_es*86400.
C
                  tot_demand_e2e3=rem_wsnde2+rem_wsnde3
                  add_supply_to_e2=amin1(avvol_to_hill,rem_capac_cs17e,
     &            tot_demand_e2e3)*frac_del_frm_wca(isa)
                  kflo(iflcs17e)=kflo(iflcs17e)+add_supply_to_e2
     &
C
                  IF (tot_demand_e2e3.gt.0.0) THEN
                    ratio_e2e3=add_supply_to_e2/tot_demand_e2e3
                  ELSE
                    ratio_e2e3=0.0
                  END IF
C
                  IF (ihwopt(is,isa).eq.0) THEN
                    avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
     &              add_supply_to_e2
                    avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
     &              add_supply_to_e2
                  END IF
                  avvol_to_hill=avvol_to_hill-add_supply_to_e2
                  rem_capac_cs17e=rem_capac_cs17e-add_supply_to_e2
                  qu(ie3)=qu(ie3)+ratio_e2e3*rem_wsnde3*86400.
                  qu(ie2)=qu(ie2)+ratio_e2e3*rem_wsnde2*86400.
C
                  add_supply_to_e1=amin1(avvol_to_hill,rem_capac_cs17w,
     &            rem_wsnde1)*frac_del_frm_wca(isa)
                  IF (ihwopt(is,isa).eq.0) THEN
                    avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-
     &              add_supply_to_e1
                    avvolrg(isa,iconvey_c)=avvolrg(isa,iconvey_c)-
     &              add_supply_to_e1
                    avvolrg(isa,iconvey_c)=amax1(avvolrg(isa,iconvey_c),
     &              0.0)
                  END IF
                  add_demand_s39=tot_demand_e2e3+rem_wsnde1
                  kflo(ifls39)=kflo(ifls39)+
```

```
     &           add_supply_to_e2+add_supply_to_e1
                IF (is39hw_opt.eq.0) THEN
                  qu(is39hw)=qu(is39hw)-(add_supply_to_e2+
     &             add_supply_to_e1)*86400.
                ELSE
                  pond(is39hw)=pond(is39hw)-(add_supply_to_e2+
     &             add_supply_to_e1)*86400./
     &             (gdar*(1.0-carfact(is39hw)-total_small_afact
     &             (is39hw)))
                END IF
                IF (is39tw_opt.eq.0) THEN
                  qu(is39tw)=qu(is39tw)+(add_supply_to_e2+
     &             add_supply_to_e1)*86400.
                ELSE
                  pond(is39tw)=pond(is39tw)+(add_supply_to_e2+
     &             add_supply_to_e1)*86400./
     &             (gdar*(1.0-carfact(is39tw)-total_small_afact
     &             (is39tw)))
                END IF
                wss39=wss39+add_supply_to_e2+
     &           add_supply_to_e1
                kflo(iflcs17w)=kflo(iflcs17w)+add_supply_to_e1
              END IF
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs17w
              istr_count=istr_count+1
              istr_count_knflw=istr_count_knflw+1
              istr_index(istr_count_knflw)=iflcs17e
            END IF
C
            IF (ihwopt(is,isa).eq.0) THEN
              qu(ihwlc)=qu(ihwlc)-cout*86400.
            ELSE
              IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &') THEN
                pond(ihwlc)=pond(ihwlc)-cout*86400./(gdar*(1.0-
     &          carfact(ihwlc)-total_small_afact(ihwlc)))
              END IF
            END IF
            qu(ie1)=qu(ie1)+wsnde1*wsfact*86400.
            qu(ie2)=qu(ie2)+wsnde2*wsfact*86400.
            qu(ie3)=qu(ie3)+wsnde3*wsfact*86400.
            qu(iese)=qu(iese)+wsndese*wsfact*86400.
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(4)) THEN
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(5)) THEN
C
C-----------------------------------------------------
C              ACME2 AND G94D (ACME IMPROVEMENT DIST)
C-----------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(2)) THEN
            pscap=dcoeff(is,isa)
            pscap=pscap*capfact(is,isa)
            IF (nint(dqu(itwlc)).gt.0.and.itwopt(is,isa).eq.0) THEN
              IF (ihwopt(is,isa).eq.0) THEN
```

```
                IF (hw .gt. floor_elev(isa,1)) THEN
                 IF (iprorte(isa).eq.1) THEN
                   ratio=avvolws(isa,iconv_canal_up_no(is,isa))/
     &              tneed(isa,iconv_canal_up_no(is,isa))
                   ratio=amin1(ratio,1.0)
                   wsneed=dqu(itwlc)*ratio
                   cout=amin1(wsneed,pscap)*wsctb*
     &              frac_del_frm_wca(isa)
                   tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &              iconv_canal_up_no(is,isa))-dqu(itwlc)
                 ELSE
                   cout=amin1(dqu(itwlc),pscap,avvolws(isa,
     &              iconv_canal_up_no(is,isa)))*wsctb*
     &              frac_del_frm_wca(isa)
                 END IF
                ELSE
                  cout = 0.0
                ENDIF
               ELSE
                 wsneed=dqu(itwlc)
                 IF (opt_pipeline_ws(ieaa_conduit_indx(is,isa)).ne.'PIPEL
     &  ') THEN
                   avail_vol_ws=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc))
     &              )*gdar/86400.
                   cout=amin1(avail_vol_ws,wsneed,pscap)*
     &              frac_del_frm_wca(isa)
                 ELSE
                   ratio=avail_vol_pipel(iserv_area_indx(is,isa))/
     &              tneed_pipel(iserv_area_indx(is,isa))
                   cout=ratio*wsneed*frac_del_frm_wca(isa)
                   tneed_pipel(iserv_area_indx(is,isa))=
     &              tneed_pipel(iserv_area_indx(is,isa))-dqu(itwlc)
                   avail_vol_pipel(iserv_area_indx(is,isa))=
     &              avail_vol_pipel(iserv_area_indx(is,isa))-cout
                 END IF
               END IF
             ELSE
               cout = 0.0
             END IF
c            print *,cout,WSNEED,WSCTB,frac_del_frm_wca(isa),hw,pscap
c    $       ,'  ACME ',rmin_stg_fc,PSCAP,AVVOLPS,chdep(itwlc),DQU(ITWLC)
c    $       ,AVVOLWS(ISA,iconv_canal_up_no(is,isa))
c    $       ,TNEED(ISA,iconv_canal_up_no(is,isa))
             kflo(iflo)=cout
C
             IF (ihwopt(is,isa).eq.0) THEN
               avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
     &          iconv_canal_up_no(is,isa))-cout
               avvolrg(isa,iconv_canal_up_no(is,isa))=amax1(avvolrg(isa,
     &          iconv_canal_up_no(is,isa)),0.0)
               avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &          iconv_canal_up_no(is,isa))-cout
             END IF
C
C-----------------------------------------------------------------
C                PROP FLOW TO STRAZZULLA WETLANDS
C-----------------------------------------------------------------
```

```
C
          ELSE IF (flnm(iflo).eq.'WC1STZ') THEN
            capacs=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            avg_stage_up=0.0
            IF (n_add_grid_loc_s(is,isa).gt.0) THEN
              DO i=1,n_add_grid_loc_s(is,isa)
                dpthgw=(ells(node_add_s(is,isa,i))-h(node_add_s(is,isa,
     &           i)))*s(node_add_s(is,isa,i))
                IF (pond(node_add_s(is,isa,i)).gt.dpthgw) THEN
                  stage_up=pond(node_add_s(is,isa,i))-dpthgw+
     &             ells(node_add_s(is,isa,i))
                ELSE
                  stage_up=(pond(node_add_s(is,isa,i))-dpthgw)/
     &             s(node_add_s(is,isa,i))+ells(node_add_s(is,isa,i))
                END IF
                avg_stage_up=avg_stage_up+stage_up/n_add_grid_loc_s(is,
     &           isa)
              END DO
            END IF
            IF (n_add_can_dep_s(is,isa).gt.0) THEN
              DO i=1,n_add_can_dep_s(is,isa)
                avg_stage_up=avg_stage_up+chdep(canal_no_add_s(is,isa,i)
     &           )/n_add_can_dep_s(is,isa)
              END DO
            END IF
            idncell=nodef(ixtw(is,isa),iytw(is,isa))
            dpthgw=(ells(idncell)-h(idncell))*s(idncell)
            IF (pond(idncell).gt.dpthgw) THEN
              stage_dn=pond(idncell)-dpthgw+ells(idncell)
            ELSE
              stage_dn=(pond(idncell)-dpthgw)/s(idncell)+ells(idncell)
            END IF
            IF (avg_stage_up.ge.stgbpt_s(is,isa,2)) THEN
              IF (stage_dn.le.stgbpt_s(is,isa,1).or.lflowws) THEN
                lflowws=.true.
                lflowws=.not.(stage_dn.gt.stgbpt_s(is,isa,2))
                IF (lflowws) THEN
                  cout=capacs
                ELSE
                  cout=0.0
                END IF
              ELSE
                cout=0.0
              END IF
            ELSE
              lflowws=.false.
              cout=0.0
            END IF
C            print *,avg_stage_up,rmxstge(is,isa),stgbpt_s(is,isa,1)
C    $     ,stgbpt_s(is,isa,2),lflowws,stage_dn,cout,capacs,'  WC1STZ'
C
C-----------------------------------------------------------------
C                     S-143  (GATED CULVERT)
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(8)) THEN
            IF (hw.gt.tw) THEN
              s143cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
```

```
            ELSE
              s143cap=0.0
            END IF
c           print *,is,isa,flnm(iflo),reg_cap_fact(is,isa,month)
c     &     ,'  S143'
C
            IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
              tw_for_constrnt=0.0
              DO kk=1,n_cells_c(is,isa)
                tw_for_constrnt=h(igrid_loc_constrnt(is,isa,kk))+
     &           pond(igrid_loc_constrnt(is,isa,kk))+tw_for_constrnt
              END DO
              tw_for_constrnt=tw_for_constrnt/n_cells_c(is,isa)
            ELSE
              tw_for_constrnt=tw
            END IF
            IF (ihwopt(is,isa).eq.0) THEN
              IF (dqu(idn_ws_cnl(is,isa,1)).gt.0.0.and.itwopt(is,isa)
     &         .eq.0) THEN
                IF (iprorte(isa).eq.1) THEN
                  ratio=avvolws(isa,iconv_canal_up_no(is,isa))/
     &             tneed(isa,iconv_canal_up_no(is,isa))
                  avail_vol_for_s34=ratio*dqu(idn_ws_cnl(is,isa,1))
                  ratio=amin1(ratio,1.0)
                  wsdel=dqu(idn_ws_cnl(is,isa,1))*ratio
                  cout=amin1(wsdel,s143cap)*frac_del_frm_wca(isa)
                  tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &             iconv_canal_up_no(is,isa))-dqu(idn_ws_cnl(is,isa,1))
                ELSE
                  avail_vol_for_s34=avvolws(isa,iconv_canal_up_no(is,
     &             isa))
                  cout=amin1(s143cap,avvolws(isa,iconv_canal_up_no(is,
     &             isa)),dqu(idn_ws_cnl(is,isa,1)))*
     &             frac_del_frm_wca(isa)
                END IF

                tdqu=dqu(idn_ws_cnl(is,isa,1))-cout
              ELSE
C
                IF (sim_opt(is,isa).eq.'REG') THEN
                  avail_vol_for_s34=avvolrg(isa,iconv_canal_up_no(is,
     &             isa))
                  IF ((trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
     &             .or.iopt_for_lokreg_flwth(is,isa).eq.1)
     &             .and.tw_for_constrnt.lt.rmxstge(is,isa,1)
     &             .and.iopt_for_reg_releases(is,isa)
     &             .eq.1.and.nint(tflg123).eq.0) THEN
                    icass=0
                    DO iwcas=1,nwcas
                      DO i=1,nstrpt(iwcas)
                        IF (flnm(istrcpt(i,iwcas))
     &                   .eq.name_struc_w_spec_code_wca(15)) THEN
                          icass=i
                          n_wca=iwcas
                          GO TO 20
                        END IF
                      END DO
```

```
                  END DO
20                IF (icass.ne.0) THEN
                    IF (ihwopt(icass,n_wca).eq.0) THEN
                      hw_s34=tw
                      avail_water=99999999.
                    ELSE
                      node=nodef(ixhw(icass,n_wca),iyhw(icass,n_wca))
                      gwdpth=(ells(node)-h(node))*s(node)
                      IF (pond(node).gt.gwdpth) THEN
                        hw_s34=pond(node)-gwdpth+ells(node)
                      ELSE
                        hw_s34=(pond(node)-gwdpth)/s(node)+ells(node)
                      END IF
                      hw_s34=amax1(hw_s34,ells(node))
                      avail_water=(hw_s34-ells(node)-
  &                    deten_def(lutyp(node)))*gdar/86400.
                    END IF
                    IF (itwopt(icass,n_wca).eq.0) THEN
                      tw_s34=chdep(icnltw(icass,n_wca))+
  &                    hdc(icnltw(icass,n_wca),iseas)
                    ELSE
                      node=nodef(ixtw(icass,n_wca),iytw(icass,n_wca))
                      gwdpth=(ells(node)-h(node))*s(node)
                      IF (pond(node).gt.gwdpth) THEN
                        tw_s34=pond(node)-gwdpth+ells(node)
                      ELSE
                        tw_s34=(pond(node)-gwdpth)/s(node)+ells(node)
                      END IF
                      tw_s34=amax1(tw_s34,ells(node))
                    END IF
                    IF (hw_s34.gt.tw_s34) THEN
                      s34cap=dcoeff(icass,n_wca)*(hw_s34-tw_s34)**
  &                    power(icass,n_wca)
                    ELSE
                      s34cap=0.0
                    END IF
                    s34cap=amin1(s34cap,avail_water)
                    IF (tw_s34 .gt. rmxstge(icass,n_wca,1)-0.5) then
                       s34cap = 1.0*s34cap*max(rmxstge(icass,n_wca,1)
  &                    +0.5-tw_s34,0.0)
                    ENDIF
                  ELSE
                    s34cap=999999.
                  END IF
                  IF (iopt_for_lokreg_flwth(is,isa)
  &                 .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,
  &                iwzone)) THEN
                    IF (rmulti_seas_pred(month)
  &                  .gt.rmulti_seas_thres_struc(is,isa)
  &                  .or.iatl_convyr_index.eq.1) THEN
                      cout=amin1(s143cap*reg_cap_fact(is,isa,month)
  &                     ,avvolrg(isa,iconv_canal_up_no(is,isa)),s34cap)
                      IF (iopt_for_lokreg_flwth(is,isa).eq.1) THEN
                        capac_lim=amin1(s143cap,s34cap)
                        cout=amin1(capac_lim,avvolws(isa,
  &                      iconv_canal_up_no(is,isa)))
                      END IF
```

```
                      ELSE
                        cout=0.0
                      END IF
C               ELSE
C                COUT = 0.0
C               ENDIF
                    ELSE
                      IF (rmulti_seas_pred(month)
     &                 .gt.rmulti_seas_thres_struc(is,isa)
     &                 .or.iatl_convyr_index.eq.1) THEN
                        cout=amin1(s143cap*reg_cap_fact(is,isa,month),
     &                  s34cap*reg_cap_fact(is,isa,month),
     &                   reg_rel_frm_lok(ieaa_conduit_indx(is,isa)),
     &                   avvolws(isa,iconv_canal_up_no(is,isa)))
                        reg_rel_frm_lok(ieaa_conduit_indx(is,isa))=
     &                   reg_rel_frm_lok(ieaa_conduit_indx(is,isa))-
     &                   cout
                      ELSE
                        cout=0.0
                      END IF
                    END IF
                  ELSE
                    avail_vol_for_s34=avvolws(isa,iconv_canal_up_no(is,
     &               isa))
                    cout=0.0
                  END IF
                ELSE
                  cout=0.0
                END IF
              END IF
c             print *,cout,trig_wca_stg(isa),regstg_wca(isa,iwzone),
c    &        tw_for_constrnt,rmxstge(is,isa,1),tflg123,s143cap,s34cap
c    &        ,'   S143F  ',dqu(idn_ws_cnl(is,isa,1))
c    &        ,avvolrg(isa,iconv_canal_up_no(is,isa))
              IF (itwopt(is,isa).eq.0.and.cnm(itwlc).eq.'L38E ') THEN
                tfls143=cout
              ELSE
                tfls143=0.0
              END IF
            ELSE
              avail_vol_up=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*
     &         gdar/86400.
              avail_vol_up=amax1(avail_vol_up,0.0)
              IF (nint(dqu(idn_ws_cnl(is,isa,1))).gt.0) THEN
                cout=amin1(s143cap,dqu(idn_ws_cnl(is,isa,1)),
     &           avail_vol_up)
              ELSE
                IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
     &           .and.tw_for_constrnt.lt.rmxstge(is,isa,1)+0.5
     &           .and.iopt_for_reg_releases(is,isa).eq.1) THEN
                 if (tw_for_constrnt .gt. rmxstge(is,isa,1)
     &            -0.5) then
                    s143cap = 1.0*s143cap*max(rmxstge(is,isa,1)
     &               +0.5 - tw_for_constrnt,0.0)
                 endif
                 cout=amin1(s143cap,avail_vol_up)
                ELSE
```

```
                cout=0.0
              END IF
            END IF
          END IF
C
          kflo(iflo)=cout
C
          IF (ihwopt(is,isa).eq.0) THEN
            avvolrg(isa,iconv_canal_up_no(is,isa))=avvolrg(isa,
     &       iconv_canal_up_no(is,isa))-cout
            avvolrg(isa,iconv_canal_up_no(is,isa))=amax1(avvolrg(isa,
     &       iconv_canal_up_no(is,isa)),0.0)
            avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &       iconv_canal_up_no(is,isa))-cout
          END IF

C
C-----------------------------------------------------------------
C                     S-141   (SHEETPILE OVERFLOW WEIR)
C----------------------------------------------------------------
        ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(26)) THEN
          IF (sim_opt(is,isa).eq.'ENV') THEN
            IF (hw.gt.tw) THEN
              s141cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              s141cap=0.0
            END IF
            IF (iopt_lim_des_cap_wca(is,isa).eq.1) THEN
              s141cap=amin1(s141cap,desgncp(is,isa))
            END IF
            s141cap=s141cap*capfact(is,isa)
            iup_targ=iup_targ_indx_wca(is,isa)
            r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
            r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
            IF (iup_targ.gt.ntotal_no_import_areas) THEN
              r1top=999.
              r1bot=998.
            END IF
            iconv_c=iconv_canal_up_no(is,isa)
            IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
              cout=s141cap
            ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
              cout=1.0/(r1top-r1bot)*s141cap*
     &         (avg_sim_wca_stg_out(iup_targ)-r1bot)
            ELSE
              cout=0.0
            END IF
            tdqu=tdqu-amin1(cout,tdqu)
          ELSE

            flshbrd=max(tw+0.10,stgbpt_s(is,isa,1))
            IF (hw.gt.flshbrd) THEN
              s141cap=dcoeff(is,isa)*(hw-flshbrd)**power(is,isa)
            ELSE
              s141cap=0.0
            END IF
            s141cap=s141cap*capfact(is,isa)
```

```
              IF (tdqu.gt.0.0.and.itwopt(is,isa).eq.0) THEN
                IF (ihwopt(is,isa).eq.1) THEN
                  avvol2bws=(hw-flshbrd)*gdar
                ELSE
                  avvol2bws=(hw-flshbrd)*cl(ihwlc)*width(ihwlc)+
     &             cgseep(ihwlc,flshbrd,0,0.)+tovflc(ihwlc,flshbrd,0,0.,
     &             0,1)
                END IF
                avvol2bws=amax1(avvol2bws/86400.,0.0)
                cout=amin1(s141cap,avvol2bws,tdqu)
                tdqu=tdqu-cout
              ELSE
                stgminrg=max(flshbrd,stgbpt_s(is,isa,2))
                IF (ihwopt(is,isa).eq.1) THEN
                  avvol2brg=(hw-stgminrg)*gdar
                ELSE
                  avvol2brg=(hw-stgminrg)*cl(ihwlc)*width(ihwlc)+
     &             cgseep(ihwlc,stgminrg,0,0.)+tovflc(ihwlc,stgminrg,0,
     &             0.,0,1)
                END IF
                avvol2brg=amax1(avvol2brg/86400.,0.0)
                IF (itwlc.eq.canal_no_add_s(is,isa,1)) THEN
                  rmaxvol=(hw-chdep(itwlc))*cl(itwlc)*width(itwlc)-
     &             cgseep(itwlc,hw,0,0.)-tovflc(itwlc,hw,0,0.,0,2)-
     &             qu(il38e)
                  rmaxvol=amax1(rmaxvol,0.0)/86400.
                  cout=amin1(s141cap,avvol2brg,rmaxvol)
                END IF
              END IF
              IF (nint(tflg123).gt.0) THEN
                cout=0.0
              END IF
            END IF
            IF (itwopt(is,isa).eq.0.and.itwlc.eq.canal_no_add_s(is,isa,
     &       1)) THEN
              tfls141=cout
            ELSE
              tfls141=0.0
            END IF
C
            kflo(iflo)=cout
C
C-----------------------------------------------------------------
C                     S-142   (GATED CULVERT)
C-----------------------------------------------------------------
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(14)) THEN
            IF (hw.gt.tw) THEN
              s142cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              s142cap=0.0
            END IF
            s142cap=s142cap*capfact(is,isa)
C
            IF (runmode.ne.'CALIB') THEN
              IF (itwopt(is,isa).eq.0) THEN
                IF (itwlc.eq.canal_no_add_s(is,isa,1)) THEN
                  wsneed=tdqu
```

```
                ELSE
                  wsneed=dqu(itwlc)
                END IF
C
                IF (nint(wsneed).gt.0) THEN
                  IF (ihwopt(is,isa).eq.0) THEN
                    IF (iprorte(isa).eq.1) THEN
                      tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &                 iconv_canal_up_no(is,isa))+tdqu
                      ratio=avvolws(isa,iconv_canal_up_no(is,isa))/
     &                 tneed(isa,iconv_canal_up_no(is,isa))
                      ratio=amin1(ratio,1.0)
                      wsdel=tdqu*ratio
                      cout=amin1(wsdel,s142cap)
                      tneed(isa,iconv_canal_up_no(is,isa))=tneed(isa,
     &                 iconv_canal_up_no(is,isa))-tdqu
                    ELSE
                      cout=amin1(s142cap,avvolws(isa,
     &                 iconv_canal_up_no(is,isa)),tdqu)
                    END IF
                  ELSE
                    avail_vol_up=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)
     &               ))*gdar/86400.
                    avail_vol_up=amax1(avail_vol_up,0.0)
                    cout=amin1(s142cap,wsneed,avail_vol_up)
                  END IF
                  kflo(iflo)=cout
                ELSE
C
                  IF (nint(tflg123).eq.0) THEN
                    IF (sim_opt(is,isa).eq.'REG') THEN
                      regc=regstg_wca(isa,iwzone)
                      IF (iopt_for_tw_constraint(is,isa).eq.1) THEN
                        tw_for_constrnt=0.0
                        DO kk=1,n_cells_c(is,isa)
                          tw_for_constrnt=h(igrid_loc_constrnt(is,isa,
     &                     kk))+pond(igrid_loc_constrnt(is,isa,kk))+
     &                     tw_for_constrnt
                        END DO
                        tw_for_constrnt=tw_for_constrnt/n_cells_c(is,
     &                   isa)
                      ELSE
                        tw_for_constrnt=tw
                      END IF
                      IF (trig_wca_stg(isa)
     &                 .gt.regc.and.tw_for_constrnt.lt.rmxstge(is,isa,1)
     &                 .and.iopt_for_reg_releases(is,isa).eq.1) THEN
                        rmaxvol=(rmxstge(is,isa,1)-chdep(il38e))*
     &                   cl(il38e)*width(il38e)-cgseep(il38e,rmxstge(is,
     &                   isa,1),0,0.)-tovflc(il38e,rmxstge(is,isa,1),0,
     &                   0.,0,2)-qu(il38e)
                        rmaxvol=amax1(rmaxvol,0.0)/86400.
                        kflo(iflo)=amin1(s142cap,rmaxvol)
                      ELSE
                        kflo(iflo)=0.
                      END IF
                    ELSE
```

```
                    kflo(iflo)=0.0
                  END IF
                ELSE
                  kflo(iflo)=0.
                END IF
              END IF
              IF (itwopt(is,isa).eq.0.and.itwlc.eq.canal_no_add_s(is,
     &         isa,1)) THEN
                tfls142=kflo(ifls142)
              ELSE
                tfls142=0.0
              END IF
            END IF
          ELSE
            kflo(iflo)=amax1(kflo(ifls34)-kflo(ifls143)-kflo(ifls141),
     &       0.0)
          END IF
          IF (ihwopt(is,isa).eq.0) THEN
            avvolws(isa,iconv_canal_up_no(is,isa))=avvolws(isa,
     &       iconv_canal_up_no(is,isa))-kflo(iflo)
          END IF
          outflow_from_3a(iconv_canal_up_no(is,isa))=
     &     outflow_from_3a(iconv_canal_up_no(is,isa))+kflo(iflo)
C
C-----------------------------------------------------------------
C                    S-34   (GATED CULVERT)
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(15)) THEN
          IF (hw.gt.tw) THEN
            s34cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
          ELSE
            s34cap=0.0
          END IF
          s34cap=s34cap*capfact(is,isa)
C
          IF (cnm(iup_canal_ws(itwlc)).eq.'C13  ') THEN
            total_need_for_s34=rem_deliv_for_nnrc
          ELSE
            total_need_for_s34=dqu(itwlc)
          END IF
C
          CALL match ('S34   ', 1, 6, struc_name_meas, nflpts, 6,
     &     imatch_h)
          IF (imatch_h.eq.0) THEN
            tfls143=kflo(ifls143)
            avail_vol_for_s34=tfls143
          END IF
          tflowup=tfls143+tfls142+tfls141
          avvol_upstream=avail_vol_for_s34+tfls142+tfls141
          IF ((total_need_for_s34.gt.0.0.and.itwopt(is,isa).eq.0)
     &     .or.tw.lt.rmxstge(is,isa,1)) THEN
            cout=amin1(tflowup,s34cap)
            IF (total_need_for_s34.gt.0.0) THEN
              wss34=amin1(cout,total_need_for_s34)
              regs34=cout-wss34
            ELSE
```

```
                wss34=0.0
                IF (iopt_for_reg_releases(is,isa).eq.1.or.nint(tflowup)
     &           .gt.0) THEN
                  regs34=cout
                ELSE
                  regs34=0.0
                  cout=0.0
                END IF
              END IF
            ELSE
              cout=0.0
              IF (iopt_for_lokreg_flwth(is,isa).eq.1) cout=
     &         amin1(tflowup,s34cap)
              wss34=0.0
              regs34=0.0
            END IF
c            print *,TFLOWUP,KFLO(IFLS143),TFLS142,TFLS141,S34CAP
c    &        ,hw,tw,COUT,'  S34  ',RMXSTGE(IS,ISA,1),imatch_h
c    &        ,iopt_for_reg_releases(is,isa)
            kflo(iflo)=cout
            rlocal_inflow=0.0
            IF (lp(23)) WRITE (62,10) ifyr+iyear-1,month,iday,
     &       flnm(ifls34),hw,tw,s34cap,avvol_upstream,cout,
     &       total_need_for_s34
            dqu(itwlc)=dqu(itwlc)-cout
            dqu(itwlc)=amax1(dqu(itwlc),0.0)
C
C
C-----------------------------------------------------------------
C                     S-151 AND S-31   (GATED CULVERTS)
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(18)) THEN
            bseflw=0.0
C -------------
C  DETERMINE MAXIMUM CAPACITY AT S-151
C -------------
            IF (ihwopt(is,isa).eq.0) THEN
              hwstg=hw+hdc(ihwlc,iseas)*0.4
            ELSE
              hwstg=hw
            END IF
C
            IF (hwstg.gt.tw) THEN
              s151cap=dcoeff(is,isa)*(hwstg-tw)**power(is,isa)
            ELSE
              s151cap=0.0
            END IF
            IF (iopt_lim_des_cap_wca(is,isa).eq.1) THEN
              s151cap=amin1(s151cap,desgncp(is,isa))
            END IF
            s151cap=s151cap*capfact(is,isa)
            iconvey_c_s151=iconv_canal_up_no(is,isa)
C
C -------------
C  DETERMINE MAXIMUM CAPACITY OF S-31
C -------------
```

```
          DO i=1,nstrpt(isa)
            IF (flnm(istrcpt(i,isa)).eq.name_struc_w_spec_code_wca(19)
  &          ) THEN
              icass=i
              GO TO 30
            END IF
          END DO
30        IF (icass.gt.0) THEN
            IF (ihwopt(icass,isa).eq.0) THEN
              hw_s31=tw
              avail_water=99999999.
            ELSE
              node=nodef(ixhw(icass,isa),iyhw(icass,isa))
              gwdpth=(ells(node)-h(node))*s(node)
              IF (pond(node).gt.gwdpth) THEN
                hw_s31=pond(node)-gwdpth+ells(node)
              ELSE
                hw_s31=(pond(node)-gwdpth)/s(node)+ells(node)
              END IF
              hw_s31=max(hw_s31,ells(node))
              avail_water=(hw_s31-ells(node)-deten_def(lutyp(node)))
  &            *gdar/86400.
            END IF
            IF (itwopt(icass,isa).eq.0) THEN
              tw_s31=chdep(icnltw(icass,isa))+hdc(icnltw(icass,isa),
  &            iseas)
            ELSE
              node=nodef(ixtw(icass,isa),iytw(icass,isa))
              gwdpth=(ells(node)-h(node))*s(node)
              IF (pond(node).gt.gwdpth) THEN
                tw_s31=pond(node)-gwdpth+ells(node)
              ELSE
                tw_s31=(pond(node)-gwdpth)/s(node)+ells(node)
              END IF
              tw_s31=max(tw_s31,ells(node))
            END IF
            IF (hw_s31.gt.tw_s31) THEN
              s31cap=dcoeff(icass,isa)*(hw_s31-tw_s31)**power(icass,
  &            isa)
            ELSE
              s31cap=0.0
            END IF
            IF (ihwopt(icass,isa).gt.0.and.itwopt(icass,isa).gt.0)
  &          THEN
              rmax_flow=(hw_s31-tw_s31)*gdar/(2.0*86400.)
            ELSE
              rmax_flow=9999999.
            END IF
            s31cap=amin1(s31cap,avail_water,rmax_flow)
          ELSE
            s31cap=0.0
          END IF
          s31cap=s31cap*capfact(is,isa)
          wsneed=0.0
          DO i=1,no_dsws_canal(is,isa)
            wsneed=wsneed+dqu(idn_ws_cnl(is,isa,i))
          END DO
```

```
            IF (ihwopt(is,isa).eq.0) THEN
              avail_vol_up=99999999.
            ELSE
              avail_vol_up=(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)))*
     &         gdar/86400.
              avail_vol_up=amax1(avail_vol_up,0.0)
            END IF
            IF (sim_opt(is,isa).eq.'REG') THEN
C -------------
C  DETERMINE REGULATORY RELEASES THROUGH S-151
C -------------
              IF (stg3a4.gt.9.5) THEN
                q=120.*(stg3a4-9.5)+1000.
              ELSE IF (stg3a4.gt.8.0) THEN
                q=667.*(stg3a4-8.0)
              ELSE
                q=0.
              END IF
              q=q*capfact(is,isa)*reg_cap_fact(is,isa,month)
C
              stage3b=trig_wca_stg(iwca3b)
              IF ((trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)
     &         .or.iopt_for_lokreg_flwth(is,isa).eq.1)
     &         .and.iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (dryseas) THEN
                  ls151=stage3b.lt.stgbpt_s(is,isa,3)
                  if (stage3b .gt. stgbpt_s(is,isa,3)-0.2) THEN
                    q = q*(stgbpt_s(is,isa,3)-stage3b)*5.0
                    if(q.lt.0.) q = 0.0
                  endif
                  IF (ls151) THEN
                    IF (iopt_for_lokreg_flwth(is,isa)
     &               .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,iwzone
     &               )) THEN
                      kflo(iflo)=amin1(q,s151cap*reg_cap_fact
     &                (is,isa,month))
                    ELSE
                      kflo(iflo)=amin1(s151cap*reg_cap_fact(is,isa
     &                 ,month),reg_rel_frm_lok(ieaa_conduit_indx(is
     &                 ,isa)),avvolws(isa,iconv_canal_up_no(is,isa)))
                    END IF
                    IF (stage3b.gt.stgbpt_s(is,isa,1)
     &              .and.tw_s31.le.rmxstge(icass,isa,1)) THEN
C
C PRESENTLY, WHEN STAGE IN WCA3B IS ABOVE 8.4 FT., S-31 IS
C OPEN TO PASS REGULATORY FLOW FROM S-151 TO C-6
C
                      IF (itwopt(icass,isa).eq.0) THEN
                         wsneed_c6=dqu(icnltw(icass,isa))
                      ELSE
                         wsneed_c6=0.0
                      END IF
                      IF (nint(wsneed).gt.0) THEN
                        ratio_ws=kflo(iflo)/wsneed
                      ELSE
                        ratio_ws=0.0
                      END IF
```

```
                  wsneed_sdcs=amax1(wsneed-wsneed_c6,0.0)
                  flow_to_c6=amax1(kflo(iflo)-ratio_ws*wsneed_sdcs,
&                  0.0)
                  flow_s31=amin1(flow_to_c6,s31cap)
                  regs31=amax1(flow_s31-wsneed_c6,0.0)
                  wss31=amin1(wsneed_c6,flow_s31)
                ELSE
                  flow_s31=bseflw
                  wss31=bseflw
                  regs31=0.0
                END IF
              ELSE
                flow_s31=bseflw
                wss31=bseflw
                regs31=0.0
                kflo(iflo)=0.0
              ENDIF
            ELSE
              IF (stage3b.ge.stgbpt_s(is,isa,2).and.ls151) THEN
                ls151=stage3b.lt.stgbpt_s(is,isa,3)
                q=q*(stgbpt_s(is,isa,3)-stage3b)*5.0
                IF (q.lt.0.) q=0.
                IF (iopt_for_lokreg_flwth(is,isa)
&                .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,
&                iwzone)) THEN
                  kflo(iflo)=amin1(q,s151cap*reg_cap_fact(is,isa
&                  ,month),avail_vol_up)
                  IF (iopt_for_lokreg_flwth(is,isa).eq.1) kflo(iflo)
&                  =amin1(q,s151cap*reg_cap_fact(is,isa,month)
&                  ,avvolws(isa,iconv_canal_up_no(is,isa)))
                ELSE
                  kflo(iflo)=amin1(s151cap*reg_cap_fact(is,isa,
&                  month),reg_rel_frm_lok(ieaa_conduit_indx(is,isa))
&                  ,avvolws(isa,iconv_canal_up_no(is,isa)))
                END IF
                IF (itwopt(icass,isa).eq.0) THEN
                  wsneed_c6=dqu(icnltw(icass,isa))
                ELSE
                  wsneed_c6=0.0
                END IF
                IF (iopt_for_reg_releases(icass,isa)
&                .eq.1.and.tw_s31.le.rmxstge(icass,isa,1)) THEN
                  IF (rmulti_seas_pred(month)
&                  .gt.rmulti_seas_thres_struc(icass,isa)
&                  .or.iatl_convyr_index.eq.1) THEN
                    IF (nint(wsneed).gt.0) THEN
                      ratio_ws=kflo(iflo)/wsneed
                    ELSE
                      ratio_ws=0.0
                    END IF
                    wsneed_sdcs=amax1(wsneed-wsneed_c6,0.0)
                    flow_to_c6=amax1(kflo(iflo)-ratio_ws*
&                    wsneed_sdcs,0.0)
                    flow_s31=amin1(flow_to_c6,s31cap)
                    regs31=amax1(flow_s31-wsneed_c6,0.0)
                    wss31=amin1(wsneed_c6,flow_s31)
                  ELSE
```

```
                  flow_s31=bseflw
                  wss31=bseflw
                END IF
              ELSE
                flow_s31=bseflw
                wss31=bseflw
              END IF
            ELSE IF (.not.ls151) THEN
              ls151=stage3b.lt.stgbpt_s(is,isa,2)
              IF (iopt_for_reg_releases(icass,isa).eq.0) THEN
                kflo(iflo)=0
                flow_s31=bseflw
                wss31=bseflw
              ELSE
                q=q*(stgbpt_s(is,isa,2)-stage3b)*2.5
                IF (q.lt.0.) q=0.
                IF (itwopt(icass,isa).eq.0) THEN
                  wsneed_c6=dqu(icnltw(icass,isa))
                ELSE
                  wsneed_c6=0.0
                END IF
                IF (iopt_for_lokreg_flwth(is,isa)
&                .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,
&                iwzone)) THEN
                  kflo(iflo)=amin1(q,s151cap*reg_cap_fact(is,isa,
&                 month),avail_vol_up)
                  IF (iopt_for_lokreg_flwth(is,isa).eq.1)
&                  kflo(iflo)=amin1(q,s151cap*reg_cap_fact(is,isa,
&                  month),avvolws(isa,iconv_canal_up_no(is,isa)))
                ELSE
                  kflo(iflo)=amin1(s151cap*reg_cap_fact(is,isa,
&                  month),reg_rel_frm_lok(ieaa_conduit_indx(is,
&                  isa)),avvolws(isa,iconv_canal_up_no(is,isa)))
                END IF
                IF (tw_s31.le.rmxstge(icass,isa,1)
&                .and.(rmulti_seas_pred(month)
&                .gt.rmulti_seas_thres_struc(icass,isa)
&                .or.iatl_convyr_index.eq.1)) THEN
                  IF (nint(wsneed).gt.0) THEN
                    ratio_ws=kflo(iflo)/wsneed
                  ELSE
                    ratio_ws=0.0
                  END IF
                  wsneed_sdcs=amax1(wsneed-wsneed_c6,0.0)
                  flow_to_c6=amax1(kflo(iflo)-ratio_ws*
&                  wsneed_sdcs,0.0)
                  flow_s31=amin1(flow_to_c6,s31cap)
                  regs31=amax1(flow_s31-wsneed_c6,0.0)
                  wss31=amin1(wsneed_c6,flow_s31)
                ELSE
                  flow_s31=bseflw
                  wss31=bseflw
                END IF
              END IF
            ELSE
              IF (iopt_for_lokreg_flwth(is,isa)
&               .eq.0.or.trig_wca_stg(isa).gt.regstg_wca(isa,
```

```
     &                iwzone)) THEN
                       kflo(iflo)=amin1(q,s151cap*reg_cap_fact(is,isa,
     &                 month),avail_vol_up)
                       IF (iopt_for_lokreg_flwth(is,isa).eq.1) kflo(iflo)
     &                  =amin1(q,s151cap*reg_cap_fact(is,isa,month),
     &                  avvolws(isa,iconv_canal_up_no(is,isa)))
                     ELSE
                       kflo(iflo)=amin1(s151cap*reg_cap_fact(is,isa,
     &                 month),reg_rel_frm_lok(ieaa_conduit_indx(is,isa))
     &                 ,avvolws(isa,iconv_canal_up_no(is,isa)))
                     END IF
                     IF (iopt_for_reg_releases(icass,isa)
     &                .eq.0.or.tw_s31.gt.rmxstge(icass,isa,1)
     &                .or.(rmulti_seas_pred(month)
     &                .le.rmulti_seas_thres_struc(icass,isa)
     &                .and.iatl_convyr_index.eq.0)) THEN
                       flow_s31=bseflw
                       wss31=bseflw
                     ELSE
                       IF (itwopt(icass,isa).eq.0) THEN
                         wsneed_c6=dqu(icnltw(icass,isa))
                       ELSE
                         wsneed_c6=0.0
                       END IF
                       IF (nint(wsneed).gt.0) THEN
                         ratio_ws=kflo(iflo)/wsneed
                       ELSE
                         ratio_ws=0.0
                       END IF
                       wsneed_sdcs=amax1(wsneed-wsneed_c6,0.0)
                       flow_to_c6=amax1(kflo(iflo)-ratio_ws*wsneed_sdcs,
     &                  0.0)
                       flow_s31=amin1(flow_to_c6,s31cap)
                       regs31=amax1(flow_s31-wsneed_c6,0.0)
                       wss31=amin1(wsneed_c6,flow_s31)
                     END IF
                   END IF
                 END IF
                 regs151=kflo(iflo)-wsneed
                 regs151=amax1(regs151,0.0)
                 wss151=amin1(kflo(iflo),wsneed)
                 IF (ihwopt(is,isa).eq.0) THEN
                   avvol_upstream=avvolws(isa,iconvey_c_s151)
                 ELSE
                   avvol_upstream=avail_vol_up
                 END IF
                 avvol_upstream_c6=flow_s31
C
               ELSE
                 wss151=0.0
                 regs151=0.0
                 IF (ihwopt(is,isa).eq.0) THEN
                   avvol_upstream=avvolws(isa,iconvey_c_s151)
                 ELSE
                   avvol_upstream=avail_vol_up
                 END IF
                 avvol_upstream_c6=bseflw
```

```
C
                kflo(iflo)=0
                flow_s31=bseflw
                wss31=bseflw
              END IF
            ELSE
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                iup_targ=iup_targ_indx_wca(is,isa)
                r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25) THEN
                    s151dsg=s151cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
     &
                    s151dsg=4.0*s151cap*(avg_sim_wca_stg_out(iup_targ)-
     &               r1top)
                  ELSE
                    s151dsg=0.0
                  END IF
                ELSE
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                    s151dsg=s151cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
     &
                    s151dsg=1.0/(r1top-r1bot)*s151cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1bot)
                  ELSE
                    s151dsg=0.0
                  END IF
                END IF
                IF (ihwopt(is,isa).eq.0) THEN
                  avvol_upstream=avvolws(isa,iconvey_c_s151)
                  kflo(iflo)=amin1(s151dsg,avvolws(isa,iconvey_c_s151))
                  avvolws(isa,iconvey_c_s151)=avvolws(isa,
     &             iconvey_c_s151)-kflo(iflo)
                  total_outf_frm_cnl(isa,iconvey_c_s151)=
     &             total_outf_frm_cnl(isa,iconvey_c_s151)+kflo(iflo)
                ELSE
                  avvol_upstream=avail_vol_up
                  kflo(iflo)=amin1(s151dsg,avail_vol_up)
                END IF
                flow_s31=0.0
                wss31=0.0
                regs151=kflo(iflo)-wsneed
                regs151=amax1(regs151,0.0)
                wss151=amin1(kflo(iflo),wsneed)
```

```
              ELSE
                regs151=0.0
                flow_s31=0.0
                wss31=0.0
                wss151=0.0
                kflo(iflo)=0.0
              END IF
C
            END IF
            outflow_from_3a(iconvey_c_s151)=
     &       outflow_from_3a(iconvey_c_s151)+kflo(iflo)
            rlocal_inflow=0.0
            IF (lp(23)) THEN
              WRITE (62,10) ifyr+iyear-1,month,iday,flnm(iflo),hw,tw,
     &         s151cap,avvol_upstream,regs151,wsneed
            END IF
C
C-----------------------------------------------------------------
C                     S-31 (GATED CULVERT)
C-----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(19)
     &     .or.flnm(iflo).eq.name_struc_w_spec_code_wca(27)) THEN
C
            IF (hw.gt.tw) THEN
              s31cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              s31cap=0.0
            END IF
            s31cap=s31cap*capfact(is,isa)
            IF (sim_opt(is,isa).eq.'ENV') THEN
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                CALL match (name_res_spec_code(13), 1, 6, resname,
     &           ntotres+1, 6, imatch)
                ilkbeltc=imatch
                IF (imatch.eq.0) ilkbeltc=ntotres+1
                iup_targ=iup_targ_indx_wca(is,isa)
                r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25) THEN
                    s31dsg=s31cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
     &
                    s31dsg=4.0*s31cap*(avg_sim_wca_stg_out(iup_targ)-
     &               r1top)
                  ELSE
```

```
                  s31dsg=0.0
                END IF
              ELSE
                IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                  s31dsg=s31cap
                ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
     &
                  s31dsg=1.0/(r1top-r1bot)*s31cap*
     &             (avg_sim_wca_stg_out(iup_targ)-r1bot)
                ELSE
                  s31dsg=0.0
                END IF
              END IF
              IF (ilkbeltc.le.ntotres) THEN
                kflo(iflo)=amin1(s31dsg,tvolwtr)
                volwtr(ilkbeltc)=volwtr(ilkbeltc)-kflo(iflo)
                tvolwtr=tvolwtr-kflo(iflo)
              ELSE
                IF (chdep(canal_no_add_s(is,isa,1)).lt.rmxstge(is,isa,
     &           1)-0.2) THEN
                  kflo(iflo)=s31dsg
                ELSE IF (chdep(canal_no_add_s(is,isa,1))
     &           .lt.rmxstge(is,isa,1)) THEN
                  kflo(iflo)=5.0*s31dsg*(rmxstge(is,isa,1)-
     &             chdep(canal_no_add_s(is,isa,1)))
                ELSE
                  kflo(iflo)=0.0
                END IF
              END IF
            ELSE
              kflo(iflo)=0.0
            END IF
          ELSE
            kflo(iflo)=flow_s31
            IF (lp(23)) THEN
              WRITE (62,10) ifyr+iyear-1,month,iday,flnm(iflo),hw,tw,
     &         s31cap,avvol_upstream_c6,kflo(iflo),dqu(itwlc)
            END IF
          END IF
C-----------------------------------------------------------------
C                     S-31 ENV2 (GATED CULVERT)
C-----------------------------------------------------------------
C
        ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(28)) THEN
C
          IF (hw.gt.tw) THEN
            s31cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
          ELSE
            s31cap=0.0
          END IF
          avail_vol=max(hw-ells(ihwlc)-deten_def(lutyp(ihwlc)),
     &     0.0)*gdar/86400.
          s31cap=s31cap*capfact(is,isa)
          IF (sim_opt(is,isa).eq.'ENV') THEN
            IF (iopt_for_reg_releases(is,isa).eq.1) THEN
              iup_targ=iup_targ_indx_wca(is,isa)
              r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,2)
```

```
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
C             print *,avg_sim_wca_stg_out(iup_targ),r1top,r1bot,' SS  '
C    $         ,stage_outflw_wca(iup_targ),offset_out(iup_targ,2)
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25) THEN
                    s31dsg=s31cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
     &
                    s31dsg=4.0*s31cap*(avg_sim_wca_stg_out(iup_targ)-
     &               r1top)
                  ELSE
                    s31dsg=0.0
                  END IF
                  s31dsg_to_bay=amin1(s31dsg,avail_vol)
                  s31dsg_to_enp=0.0
                ELSE
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                    s31dsg=s31cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot) THEN
     &
                    s31dsg=1.0/(r1top-r1bot)*s31cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1bot)
                  ELSE
                    s31dsg=0.0
                  END IF
                  s31dsg_to_bay=0.0
                  s31dsg_to_enp=amin1(s31dsg,avail_vol)
                END IF
              ELSE
                s31dsg=0.0
                s31dsg_to_bay=0.0
                s31dsg_to_enp=0.0
              END IF
            ELSE
              s31dsg=0.0
              s31dsg_to_bay=0.0
              s31dsg_to_enp=0.0
            END IF
            kflo(iflo)=s31dsg_to_enp
            qu(canal_no_add_s(is,isa,1))=qu(canal_no_add_s(is,isa,1))+
     &       s31dsg_to_bay*86400.
            pond(ihwlc)=pond(ihwlc)-s31dsg_to_bay*86400./(gdar*(1.0
     &      - carfact(ihwlc) - total_small_afact(ihwlc)))
C
C-----------------------------------------------------------------
C                     S-339   (GATED SHEETPILE BARRIER DAM)
C-----------------------------------------------------------------
```

```
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(16)) THEN
            dpthgw=(ells(ig3a2)-h(ig3a2))*s(ig3a2)
            IF (pond(ig3a2).gt.dpthgw) THEN
              stg3a2=pond(ig3a2)-dpthgw+ells(ig3a2)
            ELSE
              stg3a2=(pond(ig3a2)-dpthgw)/s(ig3a2)+ells(ig3a2)
            END IF
C
            dpthgw=(ells(ig3a9)-h(ig3a9))*s(ig3a9)
            IF (pond(ig3a9).gt.dpthgw) THEN
              stg3a9=pond(ig3a9)-dpthgw+ells(ig3a9)
            ELSE
              stg3a9=(pond(ig3a9)-dpthgw)/s(ig3a9)+ells(ig3a9)
            END IF
C
            del3239=stg3a2-stg3a9
            sch3a2=s340sch(month,iday)
            IF (hw.gt.tw) THEN
              s339cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              s339cap=0.0
            END IF
            s339cap=s339cap*capfact(is,isa)
C
            IF (ihwopt(is,isa).eq.0) THEN
              avol339=(chdep(ihwlc)-tw-0.15)*cl(ihwlc)*width(ihwlc)+
     &         cgseep(ihwlc,tw+0.15,0,0.)+tovflc(ihwlc,tw+0.15,0,0.,0,1)
     &         +kflo(ifls8)*86400.
            ELSE
              avol339=(hw-ells(ihwlc)-deten(ihwlc))*gdar
            END IF
            avol339=avol339/86400.
            avol339=amax1(avol339,0.0)
            DO i=1,nstrpt(isa)
              IF (flnm(istrcpt(i,isa)).eq.name_struc_w_spec_code_wca(18)
     &         ) THEN
               icass=i
               GO TO 40
              END IF
            END DO
   40       IF (chdep(icnlhw(icass,isa))+0.4*hdc(icnlhw(icass,isa),
     &       iseas).gt.chdep(icnltw(icass,isa))+hdc(icnltw(icass,isa),
     &       iseas)) THEN
              s151cap_lim=dcoeff(icass,isa)*(chdep(icnlhw(icass,isa))+
     &         0.4*hdc(icnlhw(icass,isa),iseas)-(chdep(icnltw(icass,isa)
     &         )+hdc(icnltw(icass,isa),iseas)))**power(icass,isa)
            ELSE
              s151cap_lim=0.0
            END IF
C
            rlkdlws=amax1(dmnd_lec_sa3-s150_wsdel,0.0)
C
            IF (rlkdlws.gt.0.0.and.itwopt(is,isa).eq.0) THEN
              kflo(iflo)=amin1(avol339,rlkdlws,s151cap_lim)
              IF (iopt_for_lokreg_flwth(is,isa)
     &         .eq.1.and.iopt_for_reg_releases(is,isa).eq.1) THEN
```

```
              IF (stg3a2.le.sch3a2) THEN
                kflo(iflo)=amin1(avol339,
     &           reg_rel_frm_lok(ieaa_conduit_indx(is,isa))+rlkdlws,
     &           s151cap_lim)
              ELSE
                IF (del3239.gt.0.13.and.del3239.le.0.8) THEN
                  q=2553.*(del3239-0.13)
                ELSE IF (del3239.gt.0.8) THEN
                  q=kflo(ifls8)*0.40
                  q=amax1(q,1711.0)
                ELSE
                  q=0.0
                END IF
                kflo(iflo)=amin1(avol339,amax1(q,rlkdlws))
              END IF
            END IF
          ELSE
            IF (iopt_for_reg_releases(is,isa).eq.1) THEN
              IF (sim_opt(is,isa).eq.'REG') THEN
                IF (iopt_for_lokreg_flwth(is,isa).eq.0) THEN
                  IF (stg3a2.gt.sch3a2) THEN
                    IF (del3239.gt.0.13.and.del3239.le.0.8) THEN
                      q=2553.*(del3239-0.13)
                    ELSE IF (del3239.gt.0.8) THEN
                      q=kflo(ifls8)*0.40
                      q=amax1(q,1711.0)
                    ELSE
                      q=0.0
                    END IF
                    q=q*capfact(is,isa)
                    IF (itwopt(is,isa).eq.0) THEN
                      tfls339=amin1(q,s339cap)
                    ELSE
                      tfls339=0.0
                    END IF
C
                    kflo(iflo)=amin1(s339cap,q)
                  ELSE
                    kflo(iflo)=0
                    tfls339=0.0
                  END IF
                ELSE
                  IF (del3239.gt.0.13.and.del3239.le.0.8) THEN
                    q=2553.*(del3239-0.13)
                  ELSE IF (del3239.gt.0.8) THEN
                    q=kflo(ifls8)*0.40
                    q=amax1(q,1711.0)
                  ELSE
                    q=0.0
                  END IF
                  q=q*capfact(is,isa)
                  IF (stg3a2.le.sch3a2) THEN
                    kflo(iflo)=amin1(s339cap,q,
     &               reg_rel_frm_lok(ieaa_conduit_indx(is,isa)))
                  ELSE
                    kflo(iflo)=amin1(s339cap,kflo(ifls8),q)
                  END IF
```

```
                tfls339=kflo(iflo)
              END IF
            ELSE
              itarg=0
              DO i=1,no_of_targets(is,isa)
                IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                  itarg=itarg+1
                END IF
              END DO
              iup_targ=iup_targ_indx_wca(is,isa)
              IF (iup_targ.le.ntotal_no_import_areas) THEN
                IF (itarg.gt.0) THEN
                  notarg=0
                  total_dpth_below_targ=0.0
                  DO i=1,no_of_targets(is,isa)
                    IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                      notarg=notarg+1
                      dpth_below_targ=
&                      stage_import_wca(itarg_indx_wca(is,isa,i))-
&                      offset(itarg_indx_wca(is,isa,i),1)-
&                      avg_sim_wca_stg(itarg_indx_wca(is,isa,i))
                      dpth_below_targ=amax1(dpth_below_targ,0.0)
                      total_dpth_below_targ=dpth_below_targ+
&                      total_dpth_below_targ
                    END IF
                  END DO
                  avg_dpth_below_targ=total_dpth_below_targ/notarg
                  struc_cap=amin1(s339cap,avol339)
                  IF (avg_dpth_below_targ.lt.0.50) THEN
                    struc_outflow=2.0*avg_dpth_below_targ*struc_cap
                  ELSE
                    struc_outflow=struc_cap
                  END IF
                  IF (avg_sim_wca_stg_out(iup_targ)
&                  .gt.stage_outflw_wca(iup_targ)+
&                  offset_out(iup_targ,2)) THEN
                    s339flw=amin1(s339cap,avol339,struc_outflow)
                  ELSE IF (avg_sim_wca_stg_out(iup_targ)
&                  .gt.stage_outflw_wca(iup_targ)) THEN
                    s339flw=1/offset_out(iup_targ,2)*amin1(s339cap,
&                    avol339)*(avg_sim_wca_stg_out(iup_targ)-
&                    stage_outflw_wca(iup_targ))
                    s339flw=amin1(s339flw,struc_outflow)
                  ELSE
                    s339flw=0.0
                  END IF
                ELSE
                  s339flw=0.0
                END IF
              ELSE
                s339flw=0.0
              END IF
              kflo(iflo)=s339flw
              tfls339=kflo(iflo)
            END IF
          ELSE
            kflo(iflo)=0.0
```

```
                tfls339=0.0
              END IF
            END IF
C
C-------------------------------------------------------------
C                     S-340   (GATED SHEETPILE BARRIER DAM)
C-------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(17)) THEN
            IF (hw.gt.tw) THEN
              s340cap=dcoeff(is,isa)*(hw-tw)**power(is,isa)
            ELSE
              s340cap=0.0
            END IF
            s340cap=s340cap*capfact(is,isa)
C
            stgmndn=max(6.0,tw+0.15)
            IF (ihwopt(is,isa).eq.0) THEN
              avol340=(chdep(ihwlc)-stgmndn)*cl(ihwlc)*width(ihwlc)+
     &         cgseep(ihwlc,stgmndn,0,0.)+tovflc(ihwlc,stgmndn,0,0.,0,1)
     &         +kflo(ifls339)*86400.
            ELSE
              avol340=(hw-ells(ihwlc)-deten(ihwlc))*gdar
            END IF
            avol340=avol340/86400.
            avol340=amax1(avol340,0.0)
C
            rlkdlws=amax1(dmnd_lec_sa3-s150_wsdel,0.0)
C
            IF (nint(rlkdlws).gt.0.and.itwopt(is,isa).eq.0) THEN
              kflo(iflo)=amin1(avol340,amax1(kflo(ifls339),s140cap),
     &         rlkdlws)
              IF (iopt_for_lokreg_flwth(is,isa).eq.1) kflo(iflo)=
     &         amin1(avol340,kflo(ifls339))
C             print *,AVOL340,RLKDLWS,WSLEC,HW,TW,MONTH,IDAY
C    &                    ,kflo(iflo)
            ELSE
              IF (sim_opt(is,isa).eq.'REG') THEN
                kflo(iflo)=amin1(s340cap,tfls339)
              ELSE
                kflo(iflo)=0.
              END IF
            END IF
C
C----------------------------------------------------------------
C                     S-346   (CULVERT ALOG L-67EXT)
C----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(20)) THEN
            IF (type_flow_across_ttrail.eq.'MINDEL ') THEN
              IF (stg3a28.lt.rmxstge(is,isa,1)) THEN
                kflo(iflo)=amin1(kflo(ifls12),desgncp(is,isa))
              ELSE
                kflo(iflo)=0
              END IF
            ELSE
              kflo(iflo)=0
```

```
            END IF
C ----------------------------------------------------------------
C                     S-345A
C----------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(29)) THEN
            avg_stage=0.0
            avg_land_elev=0.0
            DO i=1,n_add_grid_loc_s(is,isa)
              dpthgw=(ells(node_add_s(is,isa,i))-h(node_add_s(is,isa,i))
     &         )*s(node_add_s(is,isa,i))
              IF (pond(node_add_s(is,isa,i)).gt.dpthgw) THEN
                stage_loc=pond(node_add_s(is,isa,i))-dpthgw+
     &           ells(node_add_s(is,isa,i))
              ELSE
                stage_loc=(pond(node_add_s(is,isa,i))-dpthgw)/
     &           s(node_add_s(is,isa,i))+ells(node_add_s(is,isa,i))
              END IF
              avg_stage=avg_stage+stage_loc/n_add_grid_loc_s(is,isa)
              avg_land_elev=avg_land_elev+ells(node_add_s(is,isa,i))/
     &         n_add_grid_loc_s(is,isa)
            END DO
            stg3b=amax1(avg_stage,avg_land_elev)
            hdiff=hw-stg3b
            IF (hdiff.gt.2.0) THEN
              s345cap=dcoeff(is,isa)
            ELSE IF (hdiff.gt.0.0) THEN
              s345cap=dcoeff(is,isa)/2.0*hdiff
            ELSE
              s345cap=0.0
            END IF
            IF (ihwopt(is,isa).eq.1) THEN
              IF (hw.gt.ells(ihwlc)) THEN
                avail_vol=(hw-ells(ihwlc))*gdar/86400.
              ELSE
                avail_vol=0.0
              END IF
              twmax=hw*0.50+max(tw,ells(itwlc))*0.50
              depth_max=twmax-max(tw,ells(itwlc))
              depth_max=max(depth_max,0.0)
              avail_vol_max=depth_max*gdar/86400.
              s345cap=amin1(s345cap,avail_vol_max,avail_vol)
            END IF
            s345cap=s345cap*capfact(is,isa)
            iconvey_c=iconv_canal_up_no(is,isa)
C
            IF (hw.gt.stg3b) THEN
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (sim_opt(is,isa).eq.'REG') THEN
                  IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)) THEN
                    q=s345cap*capfact(is,isa)
                  ELSE IF (trig_wca_stg(isa).gt.(regstg_wca(isa,iwzone)-
     &             0.5)) THEN
                    q=s345cap*(trig_wca_stg(isa)-(regstg_wca(isa,iwzone)
     &               -0.5))*2.0*capfact(is,isa)
                  ELSE
                    q=0.0
```

```
                END IF
              ELSE
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
                iup_targ=iup_targ_indx_wca(is,isa)
                r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           2)
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25)
     &             THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top)
     &             THEN
                    s345dsg=4.0*s345cap*(avg_sim_wca_stg_out(iup_targ)
     &               -r1top)
                  ELSE
                    s345dsg=0.0
                  END IF
                ELSE
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot)
     &             THEN
                    s345dsg=1.0/(r1top-r1bot)*s345cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1bot)
                  ELSE
                    s345dsg=0.0
                  END IF
                END IF
                q=s345dsg
              END IF
            ELSE
              q=0.0
            END IF
          ELSE
            q=0.0
          END IF
C
          IF (ihwopt(is,isa).eq.0) THEN
            q=amin1(q,avvolws(isa,iconvey_c))
            avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-amin1(q,
     &       s345cap)
            total_outf_frm_cnl(isa,iconvey_c)=total_outf_frm_cnl(isa,
     &       iconvey_c)+amin1(q,s345cap)
            outflow_from_3a(iconvey_c)=outflow_from_3a(iconvey_c)+
     &       kflo(iflo)
          END IF
```

```
            kflo(iflo)=amin1(q,s345cap)
C ---------------------------------------------------------------
C                     S-345B
C---------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(30)) THEN
            avg_stage=0.0
            avg_land_elev=0.0
            DO i=1,n_add_grid_loc_s(is,isa)
              dpthgw=(ells(node_add_s(is,isa,i))-h(node_add_s(is,isa,i))
     &         )*s(node_add_s(is,isa,i))
              IF (pond(node_add_s(is,isa,i)).gt.dpthgw) THEN
                stage_loc=pond(node_add_s(is,isa,i))-dpthgw+
     &           ells(node_add_s(is,isa,i))
              ELSE
                stage_loc=(pond(node_add_s(is,isa,i))-dpthgw)/
     &           s(node_add_s(is,isa,i))+ells(node_add_s(is,isa,i))
              END IF
              avg_stage=avg_stage+stage_loc/n_add_grid_loc_s(is,isa)
              avg_land_elev=avg_land_elev+ells(node_add_s(is,isa,i))/
     &         n_add_grid_loc_s(is,isa)
            END DO
            stg3b=amax1(avg_stage,avg_land_elev)
            hdiff=hw-stg3b
            IF (hdiff.gt.2.0) THEN
              s345cap=dcoeff(is,isa)
            ELSE IF (hdiff.gt.0.0) THEN
              s345cap=dcoeff(is,isa)/2.0*hdiff
            ELSE
              s345cap=0.0
            END IF
            IF (ihwopt(is,isa).eq.1) THEN
              IF (hw.gt.ells(ihwlc)) THEN
                avail_vol=(hw-ells(ihwlc))*gdar/86400.
              ELSE
                avail_vol=0.0
              END IF
              twmax=hw*0.50+max(tw,ells(itwlc))*0.50
              depth_max=twmax-max(tw,ells(itwlc))
              depth_max=max(depth_max,0.0)
              avail_vol_max=depth_max*gdar/86400.
              s345cap=amin1(s345cap,avail_vol_max,avail_vol)
            END IF
            s345cap=s345cap*capfact(is,isa)
            iconvey_c=iconv_canal_up_no(is,isa)
C
            IF (hw.gt.stg3b) THEN
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (sim_opt(is,isa).eq.'REG') THEN
                  IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)) THEN
                    q=s345cap*capfact(is,isa)
                  ELSE IF (trig_wca_stg(isa).gt.(regstg_wca(isa,iwzone)-
     &             0.5)) THEN
                    q=s345cap*(trig_wca_stg(isa)-(regstg_wca(isa,iwzone)
     &               -0.5))*2.0*capfact(is,isa)
                  ELSE
                    q=0.0
```

```
                END IF
              ELSE
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
                iup_targ=iup_targ_indx_wca(is,isa)
                r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           2)
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25)
     &             THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top)
     &             THEN
                    s345dsg=4.0*s345cap*(avg_sim_wca_stg_out(iup_targ)
     &               -r1top)
                  ELSE
                    s345dsg=0.0
                  END IF
                ELSE
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot)
     &             THEN
                    s345dsg=1.0/(r1top-r1bot)*s345cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1bot)
                  ELSE
                    s345dsg=0.0
                  END IF
                END IF
                q=s345dsg
              END IF
            ELSE
              q=0.0
            END IF
          ELSE
            q=0.0
          END IF
C
          IF (ihwopt(is,isa).eq.0) THEN
            q=amin1(q,avvolws(isa,iconvey_c))
            avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-amin1(q,
     &       s345cap)
            total_outf_frm_cnl(isa,iconvey_c)=total_outf_frm_cnl(isa,
     &       iconvey_c)+amin1(q,s345cap)
            outflow_from_3a(iconvey_c)=outflow_from_3a(iconvey_c)+
     &       kflo(iflo)
          END IF
```

```
            kflo(iflo)=amin1(q,s345cap)
C ---------------------------------------------------------------
C                     S-345C
C---------------------------------------------------------------
C
          ELSE IF (flnm(iflo).eq.name_struc_w_spec_code_wca(31)) THEN
            avg_stage=0.0
            avg_land_elev=0.0
            DO i=1,n_add_grid_loc_s(is,isa)
              dpthgw=(ells(node_add_s(is,isa,i))-h(node_add_s(is,isa,i))
     &         )*s(node_add_s(is,isa,i))
              IF (pond(node_add_s(is,isa,i)).gt.dpthgw) THEN
                stage_loc=pond(node_add_s(is,isa,i))-dpthgw+
     &           ells(node_add_s(is,isa,i))
              ELSE
                stage_loc=(pond(node_add_s(is,isa,i))-dpthgw)/
     &           s(node_add_s(is,isa,i))+ells(node_add_s(is,isa,i))
              END IF
              avg_stage=avg_stage+stage_loc/n_add_grid_loc_s(is,isa)
              avg_land_elev=avg_land_elev+ells(node_add_s(is,isa,i))/
     &         n_add_grid_loc_s(is,isa)
            END DO
            stg3b=amax1(avg_stage,avg_land_elev)
            hdiff=hw-stg3b
            IF (hdiff.gt.2.0) THEN
              s345cap=dcoeff(is,isa)
            ELSE IF (hdiff.gt.0.0) THEN
              s345cap=dcoeff(is,isa)/2.0*hdiff
            ELSE
              s345cap=0.0
            END IF
            IF (ihwopt(is,isa).eq.1) THEN
              IF (hw.gt.ells(ihwlc)) THEN
                avail_vol=(hw-ells(ihwlc))*gdar/86400.
              ELSE
                avail_vol=0.0
              END IF
              twmax=hw*0.50+max(tw,ells(itwlc))*0.50
              depth_max=twmax-max(tw,ells(itwlc))
              depth_max=max(depth_max,0.0)
              avail_vol_max=depth_max*gdar/86400.
              s345cap=amin1(s345cap,avail_vol_max,avail_vol)
            END IF
            s345cap=s345cap*capfact(is,isa)
            iconvey_c=iconv_canal_up_no(is,isa)
C
            IF (hw.gt.stg3b) THEN
              IF (iopt_for_reg_releases(is,isa).eq.1) THEN
                IF (sim_opt(is,isa).eq.'REG') THEN
                  IF (trig_wca_stg(isa).gt.regstg_wca(isa,iwzone)) THEN
                    q=s345cap*capfact(is,isa)
                  ELSE IF (trig_wca_stg(isa).gt.(regstg_wca(isa,iwzone)-
     &             0.5)) THEN
                    q=s345cap*(trig_wca_stg(isa)-(regstg_wca(isa,iwzone)
     &               -0.5))*2.0*capfact(is,isa)
                  ELSE
                    q=0.0
```

```
                END IF
              ELSE
                itarg=0
                DO i=1,no_of_targets(is,isa)
                  IF (lwslo(itarg_indx_wca(is,isa,i))) THEN
                    itarg=itarg+1
                  END IF
                END DO
                iup_targ=iup_targ_indx_wca(is,isa)
                r1top=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           2)
                r1bot=stage_outflw_wca(iup_targ)+offset_out(iup_targ,
     &           1)
                IF (iup_targ.gt.ntotal_no_import_areas) THEN
                  r1top=999.
                  r1bot=998.
                END IF
                IF (itarg.eq.0) THEN
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top+0.25)
     &             THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1top)
     &             THEN
                    s345dsg=4.0*s345cap*(avg_sim_wca_stg_out(iup_targ)
     &               -r1top)
                  ELSE
                    s345dsg=0.0
                  END IF
                ELSE
                  IF (avg_sim_wca_stg_out(iup_targ).gt.r1top) THEN
                    s345dsg=s345cap
                  ELSE IF (avg_sim_wca_stg_out(iup_targ).gt.r1bot)
     &             THEN
                    s345dsg=1.0/(r1top-r1bot)*s345cap*
     &               (avg_sim_wca_stg_out(iup_targ)-r1bot)
                  ELSE
                    s345dsg=0.0
                  END IF
                END IF
                q=s345dsg
              END IF
            ELSE
              q=0.0
            END IF
          ELSE
            q=0.0
          END IF
C
          IF (ihwopt(is,isa).eq.0) THEN
            q=amin1(q,avvolws(isa,iconvey_c))
            avvolws(isa,iconvey_c)=avvolws(isa,iconvey_c)-amin1(q,
     &       s345cap)
            total_outf_frm_cnl(isa,iconvey_c)=total_outf_frm_cnl(isa,
     &       iconvey_c)+amin1(q,s345cap)
            outflow_from_3a(iconvey_c)=outflow_from_3a(iconvey_c)+
     &       kflo(iflo)
          END IF
```

```
            kflo(iflo)=amin1(q,s345cap)
          ENDIF
        END IF
C
        IF (flnm(iflo).ne.name_struc_w_spec_code_wca(7)) THEN
          jstr=jstr+1
          strflw(jstr,idss)=kflo(iflo)
          IF (icode_s(is,isa).eq.'GEN') THEN
            jstr=jstr+1
            strflw(jstr,idss)=env_thru_struc(is,isa)
            jstr=jstr+1
            strflw(jstr,idss)=reg_thru_struc(is,isa)
            jstr=jstr+1
            strflw(jstr,idss)=ws_thru_struc(is,isa)
          END IF
        END IF
C
C
C   store flows for local pumps into lwdd canals
C   into str2x2.dss file
C
   50 CONTINUE
C
      IF (DAY1) THEN
         do kk = 1,istr_count_knflw
           IF (flnm(istr_index(kk)) .ne. fnm(kk)) THEN
                WRITE(*,111) flnm(istr_index(kk)),fnm(kk)
 111            FORMAT(2(/),'STRUC NAME',2x,A6,2x,'WHICH IS INPUT FOR ME
     $ASURED FLOW IN model_definition_info.dat FILE OR FOR SIMULATED FLO
     $WS IN gen_nodal_dep_struc.dat AND/OR in wca_out_struc_specs.dat FI
     $LE DOES NOT MATCH NAME,',2x,A6,2x,'IN known_flow_route_specs.dat F
     $ILE. HAVE A MISMATCH;CHECK FILES.')
                STOP
           ENDIF
         enddo
      ENDIF
      IF (runmode.eq.'SIMUL') THEN
        jstr=jstr+1
        strflw(jstr,idss)=kflo(ifls39)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs2)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs4)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs9)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs12)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs17w)
        jstr=jstr+1
        strflw(jstr,idss)=kflo(iflcs17e)
      END IF
      RETURN
      END
```

wcas_input_data.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: wcas_input_data.F,v 1.16 2005/06/10 13:29:23 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wcas_input_data.F,v $
C   $Name: rel-5-5 $
C
C
C   THIS SUBROUTINE INPUTS DATA TO SIMULATE THE OUTLET STRUCTURES
C   FLOWS FROM THE WATER CONSERVATION AREAS. STRUCTURE NAMES ARE
C   SET UP FOR DSS OUTPUT.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE wcas_input_data (ds_convey_c_name,
     & int_cnl_name_for_lwdd, ds_int_canal_name_l8, idn_wca_name,
     & idn_wca_name_env, idowns_wca_name_lwdd, idn_wca_name_l8,
     & nconvey_total, convey_canal_names, rinit_wca_stage, n_conv_tot,
     & import_area_name, eaa_conv_canal_names, ntotal_no_import_areas,
     & istr_count, g94abcapac, regcoffset, iopt_lim_des_cap_wca,
     & isdnpflw_for_slope_wca, isuppflw_for_slope_wca,
     & reg_cap_fact)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
      INCLUDE 'c2.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
```

```
      INCLUDE 'environ.inc'
      INCLUDE 'wier.inc'
      INCLUDE 'trig_lec_et_param.inc'
      INCLUDE 'lec_et.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'climvar.inc'
      INCLUDE 'minfllvl.inc'
      INCLUDE 'canalloc.inc'
      INCLUDE 'dlycslope.inc'

      DIMENSION icol_add(max_n_cells_add_s),
     & icol_constrnt(max_n_cells_constrt), icol_stg_floor(5),
     & icol_wca_loc(max_n_cell_loc_wca),
     & icolposdn_wcaot(max_n_ostruc_wca,max_n_wcas),
     & icolposup_wcaot(max_n_ostruc_wca,max_n_wcas),
     & iopt_lim_des_cap_wca(max_n_ostruc_wca,max_n_wcas),
     & irow_add(max_n_cells_add_s), irow_constrnt(max_n_cells_constrt),
     & irow_stg_floor(5), irow_wca_loc(max_n_cell_loc_wca),
     & irowposdn_wcaot(max_n_ostruc_wca,max_n_wcas),
     & irowposup_wcaot(max_n_ostruc_wca,max_n_wcas),
     & isdnpflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
     & isuppflw_for_slope_wca(max_n_ostruc_wca,max_n_wcas),
     & n_conv_tot(max_n_wcas),reg_cap_red_fact(12),
     & nodepos_dnwcaot(max_n_ostruc_wca,max_n_wcas),
     & nodepos_upwcaot(max_n_ostruc_wca,max_n_wcas),
     & regcoffset(max_n_wcas), rinit_wca_stage(max_n_wcas),
     & reg_cap_fact(max_n_ostruc_wca,max_n_wcas,12)

      CHARACTER add_can_dep_id(10)*5, add_str_dep_id(10)*6,
     & cisacnl(max_n_str_inf_wca)*5, cistrindx(max_n_str_inf_wca)*6,
     & convey_canal_names(max_n_tot_conv_cnl_wcas)*5,
     & days_of_wk_wcadel(7)*3, days_week(7)*3, dnstrm_wca_name*5,
     & down_res_name*6, ds_convey_c_name(max_n_eaa_condt)*5,
     & ds_int_canal_name_l8*5, dsws_canal_name(5)*5,
     & eaa_conv_canal_names(max_n_eaa_condt)*5, iconv_canal_up_name*5,
     & iconvey_canal_no_wca_name(max_n_wcas)*5,
     & idn_canal_name(max_n_str_node_dep)*5,wca_struc_o_name(100)*6,
     & idn_wca_name(max_n_eaa_condt)*5,str_id_cap_red*6,
     & idn_wca_name_env(max_n_eaa_condt)*5, idn_wca_name_l8*5,
     & idowns_wca_name_lwdd*5, ieaa_conduit_name*5,
     & import_area_name(max_n_targ_area)*5, int_cnl_name_for_lwdd*5,
     & intcnl_name(max_n_conv_cnl_wca)*5, itarg_name(max_n_targ_loc)*5,
     & iup_canal_name(max_n_str_node_dep)*5,
     & struc_name_sim_wca(max_n_ostruc_wca,max_n_wcas)*6,
     & up_targ_name*5, wca_name_order_sim(max_n_wcas)*5

      DATA days_week /'SUN', 'MON', 'TUE', 'WED', 'THU', 'FRI', 'SAT'/

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  INPUT DATA FOR WCAOUT SUBROUTINE

      READ (21,'(I5,10(2x,A5))') NWCAS,(wca_name(ISA),ISA = 1,NWCAS)
      READ (21,*) (rinit_wca_stage(i),i=1,nwcas)
      READ (21,*) (ibasin_no_wca(i),i=1,nwcas)
      READ (21,'(5X,5(A5,2x))') (wca_name_order_sim(isa),isa=1,nwcas)
      DO isa = 1,nwcas
```

```
         CALL match (wca_name_order_sim(isa),1,5,wca_name,nwcas,5,
     $   imatch)
         IF (imatch .EQ. 0) THEN
            WRITE(*,10) wca_name_order_sim(isa)
         ENDIF
         ipwca(isa) = imatch
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      READ (21,'(5(A5,2x))') (wcaenv(i),i=1,nwcas)
      READ (21,*) (rmulti_seas_thres_chg_sched(isa),isa = 1,nwcas)
      nconvey_total = 0
      DO isa = 1,nwcas
         READ (21,'(i5,10(2x,a5,2x,F6.2))') n_intcnl(isa),(INTCNL_NAME
     $   (I) ,rmnstg(isa,i),I = 1,n_intcnl(isa))
         DO i = 1,n_intcnl(isa)
            nconvey_total = nconvey_total + 1
            CALL match (INTCNL_NAME(I),1,5,cnm,nch+1,5,imatch)
            convey_canal_names(nconvey_total) = intcnl_name(i)
            IF (imatch .EQ. 0) THEN
               WRITE(*,40) INTCNL_NAME(I)
               STOP
            ENDIF
            intcnl(isa,i) = imatch
         END DO
         n_conv_tot(isa) = nconvey_total
         READ (21,*) no_of_days_wk_wcadel(isa,1),(days_of_wk_wcadel(k)
     $   ,k = 1,no_of_days_wk_wcadel(isa,1)),frac_wcaws_del(isa,1)
         DO k = 1,no_of_days_wk_wcadel(isa,1)
            call match(days_of_wk_wcadel(k),1,3,days_week,7,3,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,50) days_of_wk_wcadel(k)
               STOP
            ENDIF
            idays_of_wk_wcadel_indx(isa,k,1) = imatch
         ENDDO
         READ (21,*) no_of_days_wk_wcadel(isa,2),(days_of_wk_wcadel(k)
     $   ,k = 1,no_of_days_wk_wcadel(isa,2)),frac_wcaws_del(isa,2)
         DO k = 1,no_of_days_wk_wcadel(isa,2)
            call match(days_of_wk_wcadel(k),1,3,days_week,7,3,imatch)
            idays_of_wk_wcadel_indx(isa,k,2) = imatch
            IF (imatch .EQ. 0) THEN
               WRITE(*,60) days_of_wk_wcadel(k)
               STOP
            ENDIF
         ENDDO
c
         IF (wca_import_variation .EQ. 'CONST') THEN
            IF (nmpts_wca(isa) .GT. 0) THEN
               avg_import_stage(isa) = 0.0
               DO k = 1,n_intcnl(isa)
                  stage_ref_env(isa,1) = 0.0
               END DO
               DO i = 1,nmpts_wca(isa)
                  avg_import_stage(isa) = avg_import_stage(isa) +
     $            rimport_stage(isa,i) / nmpts_wca(isa)
                  DO k = 1,n_intcnl(isa)
                     stage_ref_env(isa,k) = stage_ref_env(isa,k) +
```

```
     $               rimport_stage(isa,i) / nmpts_wca(isa)
                  END DO
               END DO
            ENDIF
         ENDIF
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
      READ (21,'(5(A5,2x))') (iconvey_canal_no_wca_name(i),i=1,nwcas)
      DO i = 1,nwcas
         IF (n_conv_tot(i) .GT. 0) THEN
            CALL match (iconvey_canal_no_wca_name(i),1, 5,
     $      convey_canal_names,n_conv_tot(i),5,imatch)
            IF (i .EQ. 1) THEN
               iconvey_canal_no_wca(i) = imatch
            ELSE
               IF (imatch .NE. 0) THEN
                  iconvey_canal_no_wca(i) = imatch - n_conv_tot(i - 1)
               ELSE
                  iconvey_canal_no_wca(i) = 0
               ENDIF
            ENDIF
         ELSE
            iconvey_canal_no_wca(i) = 0
         ENDIF
      END DO
c
c  READ marsh triggers for Min Flows and Levels
c
      READ (21,*) min_consec_days_bel_strtstg
      DO isa = 1,nwcas
         READ (21,*) no_of_subareas_floor(isa)
         DO i = 1,no_of_subareas_floor(isa)
            READ (21,30) avg_wca_subarea_stg_strt_count(isa,i) ,
     $      avg_wca_subarea_crit_stage(isa,i) ,
     $      avg_wca_subarea_stg_disc_lok(isa,i) ,nmpts_wca_floor(isa,i),
     $      (icol_stg_floor(j) ,irow_stg_floor(j),gage_name(isa,i,j) ,j
     $      = 1,nmpts_wca_floor(isa,i))
            DO j = 1,nmpts_wca_floor(isa,i)
               icell_no_floor(isa,i,j) = icol_stg_floor(j) - minx
     $         (irow_stg_floor(j)) + 1 + isum(irow_stg_floor(j))
            END DO
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      DO i = 1,neaacnl + 1
         CALL match(ds_convey_c_name(i),1,5,convey_canal_names ,
     $   nconvey_total,5,imatch)
         DO isa = 1,nwcas
            IF (imatch .LE. n_conv_tot(isa)) THEN
               IF (isa .GT. 1) THEN
                  iconvey_canal_no(i) = imatch - n_conv_tot(isa - 1)
               ELSE
                  iconvey_canal_no(i) = imatch
               ENDIF
               GO TO 45
            ENDIF
```

```
         END DO
   45    CONTINUE
      END DO
      CALL match(int_cnl_name_for_lwdd,1,5,convey_canal_names ,
     $nconvey_total,5,imatch)
      DO isa = 1,nwcas
         IF (imatch .LE. n_conv_tot(isa)) THEN
            IF (isa .GT. 1) THEN
               iconvey_canal_no_lwdd = imatch - n_conv_tot(isa - 1)
            ELSE
               iconvey_canal_no_lwdd = imatch
            ENDIF
            GO TO 55
         ENDIF
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
   55 CALL match(ds_int_canal_name_l8,1,5,convey_canal_names ,
     $nconvey_total,5,imatch)
      DO isa = 1,nwcas
         IF (imatch .LE. n_conv_tot(isa)) THEN
            IF (isa .GT. 1) THEN
               iconvey_canal_no_l8 = imatch - n_conv_tot(isa - 1)
            ELSE
               iconvey_canal_no_l8 = imatch
            ENDIF
            GO TO 65
         ENDIF
      END DO
c
   65 DO i = 1,nwcas
         READ (21,*) n_gage_loc(i),(icol_wca_loc(j),irow_wca_loc(j) ,j
     $   = 1,n_gage_loc(i))
         DO j=1,n_gage_loc(i)
            icell_no_wca_loc(i,j) = icol_wca_loc(j) - minx(irow_wca_loc
     $      (j)) + 1 + isum(irow_wca_loc(j))
         END DO
      END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
      DO i = 1,neaacnl + 1
         CALL match(idn_wca_name(i),1,5,wca_name,nwcas,5,imatch)
         idn_wca_indx(i) = imatch
      END DO
      DO i = 1,neaacnl
         CALL match(idn_wca_name_env(i),1,5,wca_name,nwcas,5,imatch)
         idn_wca_indx_env(i) = imatch
      END DO
      CALL match (idowns_wca_name_lwdd,1,5,wca_name,nwcas,5,imatch)
      idn_wca_name_lwdd_indx = imatch
      CALL match (idn_wca_name_l8,1,5,wca_name,nwcas,5,imatch)
      idn_wca_name_l8_indx = imatch
C
      iss = ncalcpt
      READ (21,'(I2,2x,40(A6,1x))') no_of_struc_spec_code ,
     $(name_struc_w_spec_code_wca(i),i=1,no_of_struc_spec_code)
c
      no_wca_out_struc = 0
```

```
      DO 240 ICA = 1,NWCAS
c
         READ(21,'(A)') WCAID(ICA)
         CALL match (wcaid(ica),1,5,wca_name,nwcas,5,imatch)
         IF (imatch .EQ. 0) THEN
            WRITE(*,90) wcaid(ica)
            STOP
         ENDIF
         isa = imatch
         DO I = 1,n_intcnl(isa)
            READ (21,70) NCLNS(ISA,I), (CISACNL (J),J = 1,NCLNS(ISA,I))
            READ (21,80) NSTRINF(ISA,I),(CISTRINDX(J),J =1,NSTRINF
     $      (ISA,I))
            DO J = 1,NCLNS(ISA,I)
               CALL match(CISACNL(j),1,5,cnm,nch + 1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,20) CISACNL(j),wca_name(isa)
                  STOP
               ENDIF
               isacnl(isa,i,j) = imatch
            END DO
            DO j = 1,NSTRINF(isa,i)
               CALL match(CISTRINDX(j),1,6,flnm,300,6,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,220) CISTRINDX(j),wca_name(isa)
                  STOP
               ENDIF
               ISTRINDX(ISA,I,J) = imatch
            END DO
         END DO
c
         READ (21,*) nstrpt(isa)
         READ (21,*) iprorte(isa)
         READ (21,*) stgmnrg(isa),regcoffset(isa)
c
c
         DO 230 IS = 1,NSTRPT(ISA)
            iss = iss + 1
            no_wca_out_struc = no_wca_out_struc + 1
c
            READ (21,'(A6,2(2x,A3),2X,I1,2x,F5.1,2I3)')
     $      struc_name_sim_wca (is,isa),icode_s(is,isa),sim_opt
     $      (is,isa) ,iopt_for_reg_releases(is,isa) ,
     $      rmulti_seas_thres_struc(is,isa),iwca_reg_zone(is,isa) ,
     $      iopt_for_lokreg_flwth(is,isa)
            IF (icode_s(is,isa).EQ.'GEN') READ(21,'(3(A6,2x),F6.1,4i5)')
     $      sname_env_dss(is,isa),sname_reg_dss(is,isa), sname_ws_dss
     $      (is,isa),offset_reg(is,isa),icolposup_wcaot(is,isa),
     $      irowposup_wcaot(is,isa),icolposdn_wcaot(is,isa),
     $      irowposdn_wcaot(is,isa)
c
            do imm = 1,12
              reg_cap_fact(is,isa,imm) = 1.0
            enddo
            wca_struc_o_name(no_wca_out_struc) =
     $       struc_name_sim_wca(is,isa)
            READ (21,*) IHWOPT(IS,ISA)
```

```
            IF (IHWOPT(IS,ISA) .EQ. 0) THEN
               READ (21,'(A5)') iup_canal_name(iss)
               CALL match (iup_canal_name(iss),1,5,cnm,nch + 1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,100) iup_canal_name(iss)
               ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               ICNLHW(IS,ISA) = imatch
               IF (ivary_slope_daily_opt(ICNLHW(IS,ISA)) .eq. 1 .AND.
     $         icode_s(is,isa) .EQ. 'GEN') THEN
                  DO IPOS = 1,NODCR(ICNLHW(IS,ISA))
                    IF (icolposup_wcaot(is,isa) .eq. XCN(ICNLHW(IS,ISA)
     $              ,IPOS) .AND. irowposup_wcaot(is,isa) .EQ. YCN
     $              (ICNLHW(IS,ISA),IPOS)) THEN
                       NODEPOS_UPWCAOT(IS,ISA) = IPOS
                       IF (NODEPOS_UPWCAOT(IS,ISA) .EQ. NODCR(ICNLHW(IS
     $                 ,ISA))) THEN
                          isuppflw_for_slope_wca(is,isa) = 1
                       ELSEIF (NODEPOS_UPWCAOT(IS,ISA).LE.mid_point_canl
     $                 (ICNLHW(IS,ISA))) THEN
                          isuppflw_for_slope_wca(is,isa) = 2
                       ELSEIF (NODEPOS_UPWCAOT(IS,ISA).GT.mid_point_canl
     $                 (ICNLHW(IS,ISA)) .AND. NODEPOS_UPWCAOT(IS,ISA)
     $                 .ne. NODCR(ICNLHW(IS,ISA))) THEN
                          isuppflw_for_slope_wca(is,isa) = 3
                       ENDIF
                       GO TO 225
                     ENDIF
                   ENDDO
                 ENDIF
 225           IF (struc_name_sim_wca(is,isa) .EQ.
     $         name_struc_w_spec_code_wca(7)) THEN
                  is39hw = ICNLHW(IS,ISA)
                  is39hw_opt = ihwopt(is,isa)
               ENDIF
            ELSE
               READ (21,*) IXHW(IS,ISA),IYHW(IS,ISA)
               IF (struc_name_sim_wca(is,isa) .EQ.
     $         name_struc_w_spec_code_wca(7)) THEN
                  is39hw = IXHW(IS,ISA) - minx(IYHW(IS,ISA)) + 1 + isum
     $            (IYHW(IS,ISA))
                  is39hw_opt = ihwopt(is,isa)
               ENDIF
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            READ (21,*) itwopt(is,isa)
            IF (ITWOPT(IS,ISA) .EQ. 0) THEN
               READ (21,'(A5)') idn_canal_name(iss)
               CALL match (idn_canal_name(iss),1,5,cnm,nch,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,110) idn_canal_name(iss)
               ENDIF
               ICNLTW(IS,ISA) = imatch
               IF (ivary_slope_daily_opt(ICNLTW(IS,ISA)) .eq. 1 .AND.
     $         icode_s(is,isa) .EQ. 'GEN') THEN
                  DO IPOS = 1,NODCR(ICNLTW(IS,ISA))
                    IF (icolposdn_wcaot(is,isa) .eq. XCN(ICNLTW(IS,ISA)
```

```
     $              ,IPOS) .AND. irowposdn_wcaot(is,isa) .EQ. YCN
     $              (ICNLTW(IS,ISA),IPOS)) THEN
                       NODEPOS_DNWCAOT(IS,ISA) = IPOS
                       IF (NODEPOS_DNWCAOT(IS,ISA) .EQ. 1) THEN
                          isdnpflw_for_slope_wca(is,isa) = 1
                       ELSEIF (NODEPOS_UPWCAOT(IS,ISA).LE.mid_point_canl
     $                 (ICNLHW(IS,ISA))) THEN
                          isdnpflw_for_slope_wca(is,isa) = 2
                       ELSEIF (NODEPOS_UPWCAOT(IS,ISA).GT.mid_point_canl
     $                 (ICNLHW(IS,ISA))) THEN
                          isdnpflw_for_slope_wca(is,isa) = 3
                       ENDIF
                       GO TO 235
                     ENDIF
                   ENDDO
                 ENDIF
 235           IF (struc_name_sim_wca(is,isa) .EQ.
     $         name_struc_w_spec_code_wca(7)) THEN
                  is39tw = ICNLTW(IS,ISA)
                  is39tw_opt = itwopt(is,isa)
               ENDIF
            ELSE
               READ (21,'(2i5)') IXTW(IS,ISA),IYTW(IS,ISA)
               IF (struc_name_sim_wca(is,isa) .EQ.
     $         name_struc_w_spec_code_wca(7)) THEN
                  is39tw = IXTW(IS,ISA) - minx(IYTW(IS,ISA)) + 1 + isum
     $            (IYTW(IS,ISA))
                  is39tw_opt = itwopt(is,isa)
               ENDIF
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            READ (21,*) iserv_area_indx(IS,ISA),ieaa_conduit_name
            CALL match(ieaa_conduit_name,1,5,eaa_conv_canal_names,
     $      neaacnl ,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,120) ieaa_conduit_name
               STOP
            ENDIF
            ieaa_conduit_indx(is,isa) = imatch
            READ (21,*) DCOEFF(IS,ISA),POWER(IS,ISA)
            READ (21,*) iopt_for_tw_constraint(is,isa)
            IF (struc_name_sim_wca(is,isa) .EQ. 'G94AB ' .OR.
     $       struc_name_sim_wca(is,isa) .EQ. 'LWDD ') THEN
               g94abcapac = DCOEFF(IS,ISA)
            ENDIF
            IF (iopt_for_tw_constraint(is,isa) .EQ. 1) THEN
               READ (21,*) n_cells_c(is,isa),(icol_constrnt(kk) ,
     $         irow_constrnt(kk),kk=1,n_cells_c(is,isa)),dnstrm_wca_name
               DO kk = 1,n_cells_c(is,isa)
                  igrid_loc_constrnt(is,isa,kk) = icol_constrnt(kk) -
     $            minx(irow_constrnt(kk)) + 1 + isum(irow_constrnt(kk))
               END DO
               CALL match(dnstrm_wca_name,1,5,wca_name,nwcas,5,imatch)
               idn_wca_trig(is,isa) = imatch
            ELSE
               IF (struc_name_sim_wca(is,isa) .EQ. 'S355  ') THEN
                  READ (21,*) no_of_ds_stg_lmits,(RMXSTGE(IS,ISA,ij) ,
```

```
     $            ij = 1,no_of_ds_stg_lmits)
               ELSE
                  READ (21,*) RMXSTGE(IS,ISA,1)
               ENDIF
               idn_wca_trig(is,isa) = nwcas + 1
            ENDIF
            READ (21,'(1x,2f6.0,i4)') DESGNCP(IS,ISA),CAPFACT(IS,ISA),
     $      iopt_lim_des_cap_wca(is,isa)
            READ (21,*) NBPT(IS,ISA),(STGBPT_S(IS,ISA,IBPT) ,IBPT=1,NBPT
     $      (IS,ISA))
            READ (21,'(i5,5(2x,A5))') no_dsws_canal(is,isa) ,
     $      (dsws_canal_name(i),i = 1,no_dsws_canal(is,isa))
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            DO i = 1,no_dsws_canal(is,isa)
               CALL match(dsws_canal_name(i),1,5,cnm ,nch + 1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,130) wca_name(isa),dsws_canal_name(i)
                  STOP
               ENDIF
               IDN_WS_CNL(IS,ISA,I) = imatch
            END DO
c
            READ (21,'(4x,A6)') down_res_name
c
            CALL match(down_res_name,1,6,resname,ntotres + 1,6,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,140) down_res_name
               STOP
            ENDIF
            idn_reserv_no(is,isa) = imatch
c
            READ (21,'(4x,A5)') up_targ_name
            READ (21,'(i5,2x,5(A5,2X))') no_of_targets(is,isa),
     $      (itarg_name (i),i = 1,no_of_targets(is,isa))
            READ (21,'(A5)') iconv_canal_up_name
c
            CALL match (iconv_canal_up_name,1, 5,convey_canal_names,
     $      n_conv_tot(isa),5,imatch)
            IF (imatch .NE. 0) THEN
               IF (isa .EQ. 1) THEN
                  iconv_canal_up_no(is,isa) = imatch
               ELSE
                  iconv_canal_up_no(is,isa) = imatch - n_conv_tot(isa-1)
               ENDIF
            ELSE IF (iconv_canal_up_name .NE. 'NOCNL') THEN
               WRITE(*,150) iconv_canal_up_name
               STOP
            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            READ (21,160) n_add_can_dep_s(is,isa),(add_can_dep_id(i) ,i
     $      = 1,n_add_can_dep_s(is,isa))
            READ (21,*) n_add_grid_loc_s(is,isa),(icol_add(i),irow_add
     $      (i),i = 1,n_add_grid_loc_s(is,isa))
            READ (21,170) n_add_str_dep_s(is,isa),(add_str_dep_id(i) ,i
     $      = 1,n_add_str_dep_s(is,isa))
C
            CALL match(up_targ_name,1,5,import_area_name ,
```

```
     $      ntotal_no_import_areas + 1,5,imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,180)up_targ_name
               STOP
            ENDIF
            iup_targ_indx_wca(is,isa) = imatch
c
            DO i = 1,no_of_targets(is,isa)
               CALL match(itarg_name(i),1,5,import_area_name ,
     $         ntotal_no_import_areas + 1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,190) itarg_name(i)
                  STOP
               ENDIF
               itarg_indx_wca(is,isa,i) = imatch
            END DO
c
            DO i = 1,n_add_can_dep_s(is,isa)
               CALL match (add_can_dep_id(i),1,5,cnm,nch + 1,5,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,200) add_can_dep_id(i)
                  STOP
               ENDIF
               canal_no_add_s(is,isa,i) = imatch
            END DO
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            DO i = 1,n_add_str_dep_s(is,isa)
               CALL match(add_str_dep_id(i),1,6,flnm,300,6,imatch)
               IF (imatch .EQ. 0) THEN
                  WRITE(*,210) add_str_dep_id(i)
                  STOP
               ENDIF
               add_dep_str_indx_s(is,isa,i) = imatch
            END DO
c
            DO i = 1,n_add_grid_loc_s(is,isa)
               node_add_s(is,isa,i) = icol_add(i) - minx(irow_add(i)) +
     $          1 + isum(irow_add(i))
            END DO
            CALL match (struc_name_sim_wca(is,isa),1,6,flnm,300,6,
     $      imatch)
            IF (imatch .EQ. 0) THEN
               WRITE(*,220) struc_name_sim_wca(is,isa),wca_name(isa)
               STOP
            ENDIF
            istrcpt(is,isa) = imatch
            IF (struc_name_sim_wca(is,isa) .EQ.
     $      name_struc_w_spec_code_wca(1)) iss = iss + 6
C
  230    CONTINUE
  240 CONTINUE
c
c  Input WCA outlet structures whose capacity for regulatory releases
c  would be reduced according to factors input here
c
      READ (21,*) no_str_cap_red
      DO  i = 1,no_str_cap_red
```

```
        READ (21,*) str_id_cap_red,(reg_cap_red_fact(ik),ik = 1,12)
        itot_wca_s = 0
        DO kk = 1,nwcas
           isa = IPWCA(kk)
           DO jk = 1,NSTRPT(isa)
             itot_wca_s = itot_wca_s + 1
             if (str_id_cap_red .eq. wca_struc_o_name(itot_wca_s))
     $       then
               DO imm = 1,12
                 reg_cap_fact(jk,isa,imm) = reg_cap_red_fact(imm)
               ENDDO
             endif
           ENDDO
        ENDDO
      ENDDO
c
      READ (21,*) (RFCFS355(i),RFCFS12(i),i = 1,2)
      READ (21,*) (RFCFS355_reg(i),RFCFS12_reg(i),i = 1,2)
      READ (21,*) ((total_frac_s12(i,j),i = 1,4),j = 1,2)
      READ (21,*) ((total_frac_s12_reg(i,j),i = 1,4),j = 1,2)
      DO month = 1,12
         READ (21,'(5X,15F6.0)') ((ftargs12(i,j,month),j = 1,4),i = 1,3)
      END DO
      READ (21,'(15F6.0)') (RFREGCF(i),i = 1,12)
      DO i = 1,4
         READ (21,'(15F6.0)') (rmin_stg_for_outf_s12(i,j),j = 1,12)
      END DO
      READ (21,*) (frac_capac_s343ab(i),i = 1,12)
      DO i = 1,5
         READ (21,*) no_cells_trig_s12s343(i),icol_s12s343,
     $   irow_s12s343 ,(rmax_stg_s12s343(i,j),j = 1,12)
         icell_loc_trig_s12s343(i) = icol_s12s343 - minx(irow_s12s343) +
     $    1 + isum(irow_s12s343)
      END DO
      CLOSE(21)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C STORE STRUCTURE NAMES FOR DSS OUTPUT
C
      is31rg_sim = 0
      DO ICA = 1,NWCAS
         ISA = IPWCA(ICA)
         DO IS = 1,NSTRPT(ISA)
            imatch_1 = 0
            DO i = 1,ncalcpt
               IF (flnm(ISTRCPT(IS,ISA)) .EQ. flnm(icalcpt(i)))
     $         imatch_1 = 1
            END DO
            IF (icode_s(is,isa) .EQ. 'GEN' .AND. flnm(istrcpt(is,isa))
     $      .NE. name_struc_w_spec_code_wca(9)) imatch_1 = 1
            IF (imatch_1 .EQ. 0) THEN
               istr_count = istr_count + 1
            ENDIF
            IF (struc_name_sim_wca(is,isa) .EQ.
     $      name_struc_w_spec_code_wca(1) .and. runmode .eq. 'SIMUL')
     $      THEN
               istr_count = istr_count + 6
```

```
            ENDIF
            IF (struc_name_sim_wca(is,isa) .EQ.
     $      name_struc_w_spec_code_wca(19)) is31rg_sim = 1
            IF (FLNM(ISTRCPT(IS,ISA)) .NE. name_struc_w_spec_code_wca(7)
     $      ) THEN
               MAXSTR = MAXSTR + 1
c         PRINT *,is,isa,flnm(istrcpt(is,isa)),maxstr,' SSS'
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
               STRNAME(MAXSTR) = FLNM(ISTRCPT(IS,ISA))
               IF (icode_s(is,isa) .EQ. 'GEN') THEN
                  MAXSTR = MAXSTR + 1
                  STRNAME(MAXSTR) = sname_env_dss(is,isa)
                  MAXSTR = MAXSTR + 1
                  STRNAME(MAXSTR) = sname_reg_dss(is,isa)
                  MAXSTR = MAXSTR + 1
                  STRNAME(MAXSTR) = sname_ws_dss(is,isa)
               ENDIF
            ENDIF
         END DO
      END DO
C
C  STRUCTURES INTO LWDD CANALS FOR DSS OUTPUT
C
      IF (runmode .EQ. 'SIMUL') THEN
         MAXSTR=MAXSTR+1
         CALL match('S39   ',1,6,flnm,300,6,imatch)
         STRNAME(MAXSTR) = FLNM(imatch)
         MAXSTR=MAXSTR+1
         CALL match('CS2   ',1,6,flnm,300,6,iflcs2)
         STRNAME(MAXSTR) = FLNM(iflcs2)
         MAXSTR=MAXSTR+1
         CALL match('CS4   ',1,6,flnm,300,6,iflcs4)
         STRNAME(MAXSTR) = FLNM(iflcs4)
         MAXSTR=MAXSTR+1
         CALL match('CS9   ',1,6,flnm,300,6,iflcs9)
         STRNAME(MAXSTR) = FLNM(iflcs9)
         MAXSTR=MAXSTR+1
         CALL match('CS12  ',1,6,flnm,300,6,iflcs12)
         STRNAME(MAXSTR) = FLNM(iflcs12)
         MAXSTR=MAXSTR+1
         CALL match('CS17W ',1,6,flnm,300,6,iflcs17w)
         STRNAME(MAXSTR) = FLNM(iflcs17w)
         MAXSTR=MAXSTR+1
         CALL match('CS17E ',1,6,flnm,300,6,iflcs17e)
         STRNAME(MAXSTR) = FLNM(iflcs17e)
      ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C STRUCTURES HANDLED IN KNFLOWS
      IF (is31rg_sim .EQ. 1 .OR. runmode .EQ. 'CALIB') THEN
         STRNAME(MAXSTR+1)='S31   '
         STRNAME(MAXSTR+2)='S32   '
         STRNAME(MAXSTR+3)='S32A  '
         MAXSTR=MAXSTR + 3
      ENDIF
c
c  input inFORMATion for weirs simulated in WCAs
```

```
c
      DO i = 1,max_ncells
         iwier_index(i) = 0
      END DO
      READ (133,*) no_of_wiers
      DO i = 1,no_of_wiers
         READ (133,*) icol_wier ,irow_wier,dcoef_wier(i),pwr_wier(i) ,
     $   crstelev_w(i),(wier_orient(i,k),k = 1,5)

         ndirct = 0
         DO k = 1,5
            IF (wier_orient(i,k) .NE. 'O') THEN
               ndirct = ndirct + 1
               READ(133,*) weir_name_wca(i,ndirct) ,
     $         icol_weir_tw_constrnt ,irow_weir_tw_constrnt ,
     $         rmax_tw_stage_wier(i,ndirct)
               node_weir_tw_constrnt(i,ndirct) = icol_weir_tw_constrnt -
     $         minx(irow_weir_tw_constrnt) + 1 + isum
     $         (irow_weir_tw_constrnt)
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            ENDIF
         END DO
         node_wr = icol_wier - minx(irow_wier) + 1 + isum(irow_wier)
         iwier_index(node_wr) = i
      END DO
      RETURN
   10 FORMAT(2(/),'UNRECOGNIZED WCA',2x,A5,2x,
     $'INPUT IN ORDER OF SIMULATION.CHECK wca_out_struc_specs.dat FILE')
   20 FORMAT(2(/),'UNRECOGNIZED DS CANAL NAME IN SERVICE AREA',2x,A5,2x
     $,'FOR  ',A5,2x,'CHECK  wca_out_struc_specs.dat FILE')
   30 FORMAT(3f6.2,i3,1x,5(2i3,2x,A6))
   40 FORMAT(2(/),'UNRECOGNIZED WCA CONVEYANCE CANAL NAME',2x,A5,2x,
     $'CHECK wca_out_struc_specs.dat FILE')
   50 FORMAT(2(/),'UNRECOGNIZED DAY OF WEEK',2x,A3,' FOR EAA DELIV DURIN
     $G NORMAL OR WET TIMES. CHECK wca_out_struc_specs.dat FILE')
   60 FORMAT(2(/),'UNRECOGNIZED DAY OF WEEK',2x,A3,' FOR EAA DELIV DURIN
     $G DRY TIMES OR WHEN LOK STAGE IS LOW. CHECK wca_out_struc_specs.da
     $t FILE')
   70 FORMAT(I5,2X,10(A5,1X))
   80 FORMAT(I5,2X,10(A6,1X))
   90 FORMAT(2(/),'UNRECOGNIZED WCA',2x,A5,2x,'CHECK wca_out_struc_specs
     $.dat FILE')
  100 FORMAT(2(/),'UNRECOGNIZED UPSTREAM CANAL NAME FOR WCAOUTLET STRUC'
     $,2x,A5,2x,'CHECK  wca_out_struc_specs.dat FILE')
  110 FORMAT(2(/),
     $'UNRECOGNIZED DOWNSTREAM CANAL NAME FOR WCAOUTLET STRUC',2x,A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  120 FORMAT(2(/),'UNRECOGNIZED EAA CONDUIT NAME',2x,A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  130 FORMAT(2(/),'UNRECOGNIZED DOWNSTREAM CANAL NAME MAINTAINED BY WCA'
     $,2x,A5,2x,'NAME OF CANAL IS',2x,A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  140 FORMAT(2(/),'UNRECOGNIZED DNSTREAM RESERV NAME FOR STRUC' ,2x,A6
     $,2x,'CHECK  wca_out_struc_specs.dat FILE')
  150 FORMAT(2(/),'UNRECOGNIZED UPSTREAM  CONV CANAL NAME' ,2x,A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  160 FORMAT(i5,10(2x,a5))
```

```
  170 FORMAT(i5,10(2x,a6))
  180 FORMAT(2(/),'UNRECOGNIZED UPSTREAM STAGE TARGET AREA ID' ,2x,A5
     $,2x,'CHECK  wca_out_struc_specs.dat FILE')
  190 FORMAT(2(/),'UNRECOGNIZED STAGE TARGET AREA ID',2x,A7,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  200 FORMAT(2(/),'UNRECOGNIZED ADD CANAL DEP ID',2x,A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  210 FORMAT(2(/),'UNRECOGNIZED ADD STRUC DEP ID',2x,A6,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
  220 FORMAT(2(/),'UNRECOGNIZED STRUC',2x,A6,2x,'FOR  ',A5,2x,
     $'CHECK  wca_out_struc_specs.dat FILE')
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
      END
```

wcasim.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: wcasim.inc,v 1.7 2003/07/03 20:08:08 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wcasim.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
C
      common /wcasim/ stgbpt_s(max_n_ostruc_wca,max_n_wcas,5),
     &floor_elev(max_n_wcas,max_n_conv_cnl_wca),
     &rmin_stg_wca(max_n_wcas,max_n_conv_cnl_wca),
     &iconvey_canal_no_wca(max_n_wcas),iconvey_canal_no(max_n_wcas),
     &idn_wca_indx(max_n_eaa_condt),idn_wca_name_l8_indx,
     &idn_wca_name_lwdd_indx,n_intcnl(max_n_wcas),
     &icell_no_wca_loc(max_n_wcas,max_n_cell_loc_wca),
     &node_add_s(max_n_ostruc_wca,max_n_wcas,max_n_cells_add_s),
     &n_add_can_dep_s(max_n_ostruc_wca,max_n_wcas),
     &n_add_grid_loc_s(max_n_ostruc_wca,max_n_wcas),
     &regstg_wca(max_n_wcas,max_n_zones_wca),trig_wca_stg(max_n_wcas),
     &rmean_lsel(max_n_wcas),iconvey_c_s151,
     &ncells_rec(max_ncnls,max_nostr_cnl),
     &n_gage_loc(max_n_cell_loc_wca),
     &no_dsws_canal(max_n_ostruc_wca,max_n_wcas),
     &iconv_canal_up_no(max_n_ostruc_wca,max_n_wcas),
     &isa_canal_indx(max_n_serv_area,max_n_ups_cnl_sa),
     &offset_reg(max_n_ostruc_wca,max_n_wcas),iconvey_canal_no_lwdd,
     &n_add_str_dep_s(max_n_ostruc_wca,max_n_wcas),iconvey_canal_no_l8,
     &capacpsac_ws,canal_no_add_s(max_n_ostruc_wca,max_n_wcas,5),
     &add_dep_str_indx_s(max_n_ostruc_wca,max_n_wcas,5),
     &iopt_for_reg_releases(max_n_ostruc_wca,max_n_wcas),node_dn_s355,
     &iup_targ_indx_wca(max_n_ostruc_wca,max_n_wcas),
     &idn_wca_indx_env(max_n_eaa_condt),ibasin_no_wca(max_n_wcas),
     &iserv_area_indx(max_n_ostruc_wca,max_n_wcas),
     &ws_pipeline_3a,ws_pipeline_2a,
     &ws_pipeline_ca1,ieaa_conduit_indx(max_n_ostruc_wca,max_n_wcas),
     &ws_pipeline_3a_s150,ws_pipeline_ca1_hill,ws_pipeline_ca1_wpb,
```

```
     &total_outf_from_holy_g200a,iroute_to_miamic_via_g200a,
     &idn_reserv_no(max_n_ostruc_wca,max_n_wcas),tvolwtr_res(max_n_res),
     &idn_wca_trig(max_n_ostruc_wca,max_n_wcas),wcaenv(max_n_wcas),
     &icode_s(max_n_ostruc_wca,max_n_wcas),
     &opt_pipeline_ws(max_n_eaa_condt)
C-------------------------------------

      CHARACTER wcaenv*5,icode_s*3,opt_pipeline_ws*7
      INTEGER canal_no_add_s,add_dep_str_indx_s
        DOUBLE PRECISION stgbpt_s,floor_elev,rmin_stg_wca
```

wier.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: wier.inc,v 1.3 2003/07/03 20:08:08 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wier.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------------------------
      common /weir/ weir_name_wca(max_n_pass_weirs,5),
     &dcoef_wier(max_n_pass_weirs),pwr_wier(max_n_pass_weirs),
     &crstelev_w(max_n_pass_weirs),wier_orient(max_n_pass_weirs,5),
     &iwier_index(max_ncells),qwier(max_n_pass_weirs,5),no_of_wiers,
     &rmax_tw_stage_wier(max_n_pass_weirs,5),
     &node_weir_tw_constrnt(max_n_pass_weirs,5)
      character weir_name_wca*6,wier_orient*1
```

wmm_par.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: wwilcox $
C     $Id: wmm_par.inc,v 1.9 2004/09/28 18:12:25 wwilcox Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wmm_par.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      parameter(max_n_seas=2,max_n_res_in_cell=3,max_n_reg_rel_basin=3,
     &max_n_cell_loc_gage=3,max_n_cells_constrt=3,max_n_substr=3,
     &max_n_wcas=5,max_add_grid_loc=5,max_no_clim_thres=5,
     &max_n_ds_cnl_lim=5,max_n_dual_cond=5,max_n_cells_rec=5,
     &max_n_dest_lvsp=5,max_n_conv_cnl_wca=5,max_n_serv_area=5,
     &max_n_cells_reuse=5,max_n_cnls_reuse=5,max_n_res_reuse=5,
     &max_n_str_inf_wca=5,max_n_runff_thres=5,max_n_asr_frm_res=5,
     &max_n_res_rec_str_flw=5,max_n_tide_outf=5,max_n_targ_loc=5,
     &max_n_targ_struc=5,max_n_eaa_condt=5,max_n_minfllvl_loc=7,
     &max_n_puls_rel=10,max_nostr_cnl=10,max_nostr_res=10,
     &max_n_lok_zones=10,max_n_stor_w_oper_sch=10,max_n_puls_zones=10,
     &max_n_zones_wca=10,max_n_cell_loc_wca=10,max_n_ups_cnl_sa=10,
     &max_n_inf_loc_str=10,max_n_out_loc_str=10,max_n_cells_rec_flw=10,
     &max_n_reach_eaa=10,max_n_bkpts=15,max_n_rows_nnrb=15,
     &max_n_strt_cnls_ws=15,max_n_loc_area=15,max_n_flwth_cnls=20,
     &max_n_sm_res=20,max_no_lvsp=20,max_n_res_grid_loc=20,
     &max_n_div_to_ecp=20,max_n_oper_lines=25,max_n_res=40,
     &max_n_ostruc_wca=30,max_n_targ_area=50,
     &max_n_str_lok_in_wca=100,max_n_cnl_dual_ops=50,max_n_str_drwdn=50,
     &max_n_str_res_drwdn=50,max_n_lu_type=50,max_n_reuse_plnts=50,
     &max_n_lp_flags=60,max_n_rows=66,max_ncnlloc=75,
     &max_n_cnls_w_var_slpe=100,max_n_year=100,
     &max_n_str_node_dep=100,max_n_str_name_spec=100,
     &max_n_res_name_spec=100,max_n_gage_loc_outp=100,
     &max_n_str_w_ds_targ=100,max_n_pass_weirs=100,max_ncnls=220,
     &max_n_lvsp_loc=200,max_n_struc_flw=400,max_n_cells_lec=550,
     &max_n_tot_cnl_loc=1000,max_n_entries_sim_flw_outp=1000,
     &max_n_months=1200,max_n_wells_rss=1200,max_n_wells=2000,
     &max_ncells=2000,max_ncells_w_lok=2250,max_n_days_sim=37000,
```

```
&max_n_ds_cell_lim=3,max_n_weir_outf=5,max_n_struc_meas=100,
&max_n_rows_incl_lok=82,max_n_goal_ssm=10,max_n_rows_lok=17,
&max_n_tot_conv_cnl_wcas=50,max_n_cells_nsm=2500,
&max_n_cells_add_s=5,max_n_cells_lok=176,max_n_stas_ecp=10,
&max_n_cnl_w_weir_seas=100,max_n_storm_drwdn=100,
&max_n_cnl_min_bel_flr=100)
```

wrout.inc

```
c DISCLAIMER:
c
c ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
c RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
c IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
c WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
c AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
c WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
c THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
c REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
c PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
c RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
c RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
c IS ENTIRELY ASSUMED BY THE RECIPIENT.
c

C-----------------------------------------------------
C
C     CVS Keywords
C     $Author: rnovoa $
C     $Id: wrout.inc,v 1.8 2004/01/15 21:27:05 rnovoa Exp $
C     $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wrout.inc,v $
C       $Name: rel-5-5 $
C
C
C-----------------------------------------------------
      common /WROUT/ RMXSTG(max_ncnls,max_nostr_cnl,max_n_seas),
     &RMNSTGS(max_ncnls,max_nostr_cnl,max_n_seas),
     &stg_trig_fc_ops(max_ncnls,max_nostr_cnl),
     &CNLSTG(max_ncnls,max_nostr_cnl,max_n_ds_cnl_lim),
     &STGINTKE(max_ncnls,max_nostr_cnl,max_n_substr,max_n_seas),
     &dpth_above_mn_cap_red(max_ncnls,max_nostr_cnl,max_n_seas),
     &rmin_l8stg_flow_to_mcnl(max_n_seas),IHLCUL,NODANGL,IFLS356,
     &LC1P,ISNCRE,DQU(max_ncnls),IDNOPT,IFLS335,IFLS349A,
     &IFLS336,IFLS337,IFLHLSBR,TFLS331,HLCNLDS,capacs5as,
     &QS18CR,IHLDS1,IHLDS2,DRAINHIL,nsfc_w_chg_trg,IFLS349B,
     &ISIMUL(max_ncnls),nmdays_rainavg,nmdays_rainavg0,
     &IRATING(max_ncnls,max_nostr_cnl,max_n_substr,max_n_seas),
     &IPURPSE(max_ncnls,max_nostr_cnl,max_n_substr),
     &ISPEC(max_ncnls,max_nostr_cnl),RMINHD(max_ncnls,max_nostr_cnl),
     &DISCOEF(max_ncnls,max_nostr_cnl,max_n_substr,max_n_seas),
     &PWR(max_ncnls,max_nostr_cnl,max_n_substr,max_n_seas),IL33,IFLS197,
     &ICNUM(max_ncnls,max_nostr_cnl,max_n_ds_cnl_lim),
     &NDSCLIM(max_ncnls,max_nostr_cnl),IFLS155,IS22UP,NNDREC(max_ncnls),
     &IRVBSN(max_ncnls,max_nostr_cnl),L29,ixfc_trg_ops(100,2),
     &IXRCV(max_ncnls,max_nostr_cnl,max_n_cells_rec_flw),
     &IYRCV(max_ncnls,max_nostr_cnl,max_n_cells_rec_flw),L31S,
     &IRCEIVE(max_ncnls,max_nostr_cnl),iyfc_trg_ops(100,2),
     &NODRCV(max_ncnls,max_nostr_cnl,max_n_cells_rec_flw),
     &STRDCAP(max_ncnls,max_nostr_cnl,max_n_substr,max_n_seas),IL37,
     &IS25BUP,IWPBCAT,IBC51W,ndays_sim_sd,grid_loc_trg_ops(100),
     &s331_pump,canal_no_add(max_n_str_node_dep,5),IFLG124,PMPTOMCNL,
     &DRAINC51W,NSUBSTR(max_ncnls,max_nostr_cnl),igrid_loc_on_off(50),
     &NCOEFF(max_ncnls,max_nostr_cnl),icnl_no_upstrm_trg_ops(100),
     &LPUMP(max_ncnls,max_nostr_cnl,max_n_substr),IC4,FLFRML8TOCA1,
```

```
     &FLFRML8TOC51W,int_cnl_holey(max_n_eaa_condt),NTIDEOUT(max_ncnls),
     &YCNLNODE(max_ncnls,max_n_tide_outf),DRAINC51,WSFRC51W,IMCNLE,
     &IG618,QS39TBR,LHLS,WSDELFRWPB,COUTC10A,igrid_loc_trg_ops(100,2),
     &NODEPOS_SU(max_ncnls,max_nostr_cnl),IFLS176,icnl_no_on_off(50),
     &NODEPOS_SD(max_ncnls,max_nostr_cnl),istruc_no_chg_trg_ops(100),
     &NODEPOS_WU(max_ncnls,max_nostr_cnl),node_pos_trg_ops(100,2),
     &NODEPOS_WD(max_ncnls,max_nostr_cnl),tseepge(max_no_lvsp),NODL8R2,
     &NODL8R1,NODG261,NODG262,NODG263,icanl_no_trg_ops(100,2),
     &add_dep_str_indx(max_n_str_node_dep,5),istruc_no_on_off(50),
     &avg_sim_wca_stg(max_n_targ_area),NODS343,NODS344,
     &rmin_diff_below_targ(max_ncnls,max_nostr_cnl),
     &rmax_diff_below_targ(max_ncnls,max_nostr_cnl),
     &outflow_from_sta6_to_s140,holy_out_to_s140,rmin_stg_l30_ws,
     &proj_sea_level_rise,iflc11r,node_pos_on_off(50),
     &rmax_wpbcat_for_inflw,iflc9r,est_flow_to_gwt(max_n_res),ilv30,
     &name_struc_w_spec_code_wca(50),strname_spec(150),
     &n_cells_c(max_n_ostruc_wca,max_n_wcas),canl_names_spec(100),
     &s331ws,s331fc,ids_trg_idx(max_ncnls,max_nostr_cnl),
     &no_cells_trig_s12s343(5),icell_loc_trig_s12s343(5),
     &LGRAV(max_ncnls,max_nostr_cnl,max_n_substr),ig3273_const_indx,
     &rinflw_to_res_frm_cnl(max_n_res),no_struc_sing_on_off,
     &stage_res(max_ncells,max_n_res_in_cell),trig_stg_on_off(50),
     &pond_in_res(max_ncells,max_n_res_in_cell),offset_on_off(50),
     &ells_in_res(max_ncells,max_n_res_in_cell),i_counter_mx(5),
     &lutyp_in_res(max_ncells,max_n_res_in_cell),
     &LPUMPF(max_ncnls,max_nostr_cnl,max_n_substr),
     &icell_no_sd(max_ncnls,max_nostr_cnl),iups_cnl_no_on_off(50),
     &icell_no_su(max_ncnls,max_nostr_cnl),i_counter_m(5),
     &itrg_loc_ops_cnl(100,2),istruc_no_alt_trg_ops(100),
     &imatch_struc_loc(max_ncnls,max_nostr_cnl),
     &imatch_struc_no(max_nostr_cnl,5),no_match_loc(max_nostr_cnl),
     &prev_outf_to_ds_cnl(max_ncnls,max_ncnls),outf_s9,
     &outf_fc_thru_s381,rmin_stg_avol(max_ncnls,max_nostr_cnl),
     &iopt_for_interdep_on_off(max_ncnls,max_nostr_cnl),
     &icount_bel_fc_thres(max_ncnls,max_nostr_cnl,max_n_substr)
C-----------------
        LOGICAL LPUMP,LGRAV,LPUMPF
        INTEGER YCNLNODE,canal_no_add,add_dep_str_indx
        REAL NNDREC
        character name_struc_w_spec_code_wca*6,strname_spec*6
     $    ,canl_names_spec*5
        DOUBLE PRECISION RMXSTG,RMNSTGS,cnlstg,rmin_l8stg_flow_to_mcnl
     $  ,STGINTKE,stg_trig_fc_ops
```

ws_from_res.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: wwilcox $
C   $Id: ws_from_res.F,v 1.19 2005/06/10 13:29:23 wwilcox Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/ws_from_res.F,v $
C   $Name: rel-5-5 $
C
C
C  purpose: compute water supply discharges from reservoir to appropriate
C  destination(s)
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE ws_from_res (irs, js, res_no, ieaabsn, crmin, sumrain,
     & jday, isem_flg, dmnd_sugh, dmnd_sugh_orig, capac_semws,
     & capac_sugh, stgref_asr_rec, res_outflow_to_basin, stagelo_beg,
     & pctmet, sem_dmnd_orig, iopt_sim_sta2_for_ws,
     & outflow_from_sta2_for_ws, wsndes, wsndese, runoff_to_sta, volwtr,
     & frac_thru_s8_sta34, wsnde1, wsnde2, wsnde3, res_outflow_env,
     & rloc_excess_for_ws, tot_flw_to_c9r_frm_nlkblt,
     & ws_from_acmres_lwdd, issm_cutb_opt_bcyp, pot_del_to_eaa_basin,
     & alocnnr_potdel, total_lec_dmnd_lok, iroute_to_wca3a_frm_holy,
     & iopt_env_eaar_thru_g372)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stas.inc'
      INCLUDE 'simflows.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'c1.inc'
```

```
      INCLUDE 'c2.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'asr_param.inc'
      INCLUDE 'asr1.inc'
      INCLUDE 'resadj.inc'
      INCLUDE 'resws.inc'
      INCLUDE 'wcasim.inc'
      INCLUDE 'loktarg.inc'
      INCLUDE 'resbudg.inc'
      INCLUDE 'smallres.inc'

      DIMENSION alocnnr_potdel(2), crmin(max_ncnls,max_n_seas),
     & pot_del_to_eaa_basin(max_n_eaa_condt,2),
     & res_outflow_to_basin(max_n_res,max_nostr_res),
     & rloc_excess_for_ws(max_n_serv_area), sumrain(max_ncells),
     & volwtr(max_n_res)

      CHARACTER struc_name_s13a(10)*6

      INTEGER res_no

      DOUBLE PRECISION crmin,wcrsth_s,dble8p5
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  compute index for condition for LOK del to Major EAA basins

      if (nint(total_lec_dmnd_lok) .gt. 200) then
         indx_lok_del = 1
      else
         indx_lok_del = 2
      endif

C  determine available volume of water available for
C  each outlet structure

      is=istrctr_ws(res_no,js)
      IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(15))
     & RETURN
      IF (sta34_outf_flex_eaar_opt.eq.'YES') THEN
        CALL match (name_res_spec_code(10), 1, 6, resname, ntotres+1, 6,
     &   ieaarsn_no)
        CALL match (name_res_spec_code(11), 1, 6, resname, ntotres+1, 6,
     &   ieaarss_no)
        DO kk=1,nmtr
          IF (ieaarsn_no.gt.0) THEN
            IF (ibsn(mtrp(kk,1)).eq.ibsn_no_res(ieaarsn_no)) THEN
              stg_eaa_res_2a=acstg(kk,1)
              stg_eaar_min_2a = ells(mtrp(kk,1))+1.0
            END IF
          END IF
          IF (ieaarss_no.gt.0) THEN
            IF (ibsn(mtrp(kk,1)).eq.ibsn_no_res(ieaarss_no)) THEN
              stg_eaa_res_2b=acstg(kk,1)
              stg_eaar_min_2b=ells(mtrp(kk,1))+1.0
            END IF
          END IF
        END DO
```

```
 END IF
 IF (res_out_strname(res_no,is).ne.res_struc_name_spec_code(16)
&.and.res_out_strname(res_no,is).ne.res_struc_name_spec_code(17))
&THEN
   IF (inodopt(res_no,is,2).eq.1) THEN
    IF (sfactor(res_no) .GE. sfactmin .or. ires_small_sim
&   (res_no) .eq. 'NO ') THEN
     depgwh = (ells(idnstrm(res_no,is))-h(idnstrm(res_no,is)))
&    *s(idnstrm(res_no,is))
     IF (pond(idnstrm(res_no,is)) .gt. depgwh) THEN
       head_g=pond(idnstrm(res_no,is))-depgwh +
&      ells(idnstrm(res_no,is))
     ELSE
       head_g = (pond(idnstrm(res_no,is))-depgwh)
&      /s(idnstrm(res_no,is)) + ells(idnstrm(res_no,is))
     ENDIF
     IF (sfactor(res_no).le.1.0) THEN
       sgn=1
     ELSE
       sgn=-1
     END IF
     adjust_for_ovlf=sum_chg_dpth_ovl_flw(idnstrm(res_no,is))
     term1_for_adjust_area=(sumet_outside_res(idnstrm(res_no,is))+
&     seep_loss_res(idnstrm(res_no,is))+
&     daily_seep_loss_res(idnstrm(res_no,is))+
&     daily_et_outside_res(idnstrm(res_no,is))-sumrain(idnstrm
&     (res_no,is))/12.0-sum_chg_depth_levee_spg(idnstrm(res_no,is))
&     )*((1.0-sfactor(res_no))/sfactor(res_no))
     term2_for_adjust_area=(outf_fc_frm_res_to_cell(idnstrm(res_no,
&     is))+outf_seep_loc_to_res_cell(idnstrm(res_no,is))+
&     daily_outf_fc_frm_res_to_cell(idnstrm(res_no,is))+
&     adjust_for_ovlf)/sfactor(res_no)
     IF (head_g.ge.ells(idnstrm(res_no,is))) THEN
       IF (sgn*term1_for_adjust_area.lt.term2_for_adjust_area)
&       THEN
         adjust_dur_excess(idnstrm(res_no,is))=
&         adjust_for_area(idnstrm(res_no,is))+
&         adjust_dur_deficit0(idnstrm(res_no,is))
         adjust_for_area1 = 0.0
       ELSE
         adjust_dur_deficit0(idnstrm(res_no,is))=
&         adjust_dur_excess(idnstrm(res_no,is))
         adjust_dur_excess(idnstrm(res_no,is))=
&         adjust_dur_deficit0(idnstrm(res_no,is))
       END IF
       head=ells(idnstrm(res_no,is))+pond(idnstrm(res_no,is))/
&       sfactor(res_no)+term1_for_adjust_area-
&       adjust_dur_excess(idnstrm(res_no,is))
&       -term2_for_adjust_area
       IF (sgn*term1_for_adjust_area.ge.term2_for_adjust_area)
&       THEN
         adjust_for_area(idnstrm(res_no,is))=term1_for_adjust_area-
&         term2_for_adjust_area+daily_excess_adjust(idnstrm(res_no,
&         is))
         IF (sfactor(res_no).le.1.0) THEN
           adjust_for_area(idnstrm(res_no,is))=
&           amax1(adjust_for_area(idnstrm(res_no,is)),0.0)
```

```
              END IF
              adjust_for_area(idnstrm(res_no,is))=sgn*amin1(sgn*
     &         adjust_for_area(idnstrm(res_no,is)),sgn*(head-
     &         (h(idnstrm(res_no,is))+pond(idnstrm(res_no,is)))))
              adjust_for_area1 = adjust_for_area(idnstrm(res_no,is))
            END IF
            head=head-adjust_for_area1
          ELSE
            head = head_g
          END IF
c         head=amax1(head,ells(idnstrm(res_no,is)))
         ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
            DO ir = 1,noresincell(idnstrm(res_no,is))
              IF (res_no .eq. ires_index_in_cell(idnstrm(res_no,is)
     &         ,ir)) THEN
                head=stage_res(idnstrm(res_no,is),ir)
              ENDIF
            ENDDO
          ENDIF
        ELSE
          head=chdep(ioutcnl(res_no,is))
        END IF
!
        IF (inodopt(res_no,is,3).eq.1) THEN
         IF (((sfactor(iwmads(res_no,is)) .GE.sfactmin.or.ires_small_sim
     &   (iwmads(res_no,is)) .eq. 'NO ').AND. iwmads(res_no,is) .le.
     &   ntotres) .OR. iwmads(res_no,is) .gt. ntotres) THEN
c  downstream reservoir is large enough not to treat as separate entity in
c  grid domain
          tail=pond(nodepw(res_no,is,1))-(ells(nodepw(res_no,is,1))-
     &     h(nodepw(res_no,is,1)))*s(nodepw(res_no,is,1))+
     &     ells(nodepw(res_no,is,1))
          IF (iwmads(res_no,is).gt.ntotres) THEN
            afact=1.0
          ELSE
            afact=sfactor(iwmads(res_no,is))
          END IF
          IF (afact.le.1.0) THEN
            sgn=1
          ELSE
            sgn=-1
          END IF
          IF (tail.gt.ells(nodepw(res_no,is,1)).and.nodepw(res_no,is,1)
     &     .ne.idnstrm(res_no,is)) THEN
            adjust_for_ovlf=sum_chg_dpth_ovl_flw(nodepw(res_no,is,1))
            term1_for_adjust_area=(sumet_outside_res(nodepw(res_no,is
     &       ,1))+seep_loss_res(nodepw(res_no,is,1))+
     &       daily_seep_loss_res(nodepw(res_no,is,1))+
     &       daily_et_outside_res(nodepw(res_no,is,1))-sumrain(nodepw
     &       (res_no,is,1))/12.0-sum_chg_depth_levee_spg(nodepw(res_no
     &       ,is,1)))*((1.0-sfactor(res_no))/sfactor(res_no))
            term2_for_adjust_area=
     &       (outf_fc_frm_res_to_cell(nodepw(res_no,is,1))+
     &       outf_seep_loc_to_res_cell(nodepw(res_no,is,1))+
     &       daily_outf_fc_frm_res_to_cell(nodepw(res_no,is,1))
```

```
     &       +adjust_for_ovlf)/sfactor(res_no)
             IF (sgn*term1_for_adjust_area.lt.term2_for_adjust_area)
     &       THEN
              adjust_dur_excess(nodepw(res_no,is,1))=
     &         adjust_for_area(nodepw(res_no,is,1))+
     &         adjust_dur_deficit0(nodepw(res_no,is,1))
             ELSE
              adjust_dur_deficit0(nodepw(res_no,is,1))=
     &         adjust_dur_excess(nodepw(res_no,is,1))
              adjust_dur_excess(nodepw(res_no,is,1))=
     &         adjust_dur_deficit0(nodepw(res_no,is,1))
             END IF
             tail=ells(nodepw(res_no,is,1))+pond(nodepw(res_no,is,1))/
     &       afact+term1_for_adjust_area-
     &       adjust_dur_excess(nodepw(res_no,is,1))-
     &       term2_for_adjust_area
             IF (sgn*term1_for_adjust_area.ge.term2_for_adjust_area)
     &       THEN
              adjust_for_area(nodepw(res_no,is,1))=
     &         term1_for_adjust_area-term2_for_adjust_area+
     &         amax1(daily_excess_adjust(nodepw(res_no,is,1)),0.0)
              tail=tail-adjust_for_area(nodepw(res_no,is,1))
             END IF
            END IF
            tail=amax1(tail,ells(nodepw(res_no,is,1)))
            wcrsth_s=max(tail,crstelev(res_no,is,2))
           ELSE
c  downstream reservoir is small ,treated a separate entity in grid domain
c
             IF (nodepw(res_no,is,1) .eq. idnstrm(res_no,is)) then
                deptgw =(ells(nodepw(res_no,is,1))-h(nodepw(res_no,is,1))
     &          ) * s(nodepw(res_no,is,1))
                IF (pond(nodepw(res_no,is,1)).gt. deptgw)
     &          THEN
                   tail = pond(nodepw(res_no,is,1))
     &             - deptgw + ells(nodepw(res_no,is,1))
                ELSE
                   tail = (pond(nodepw(res_no,is,1))
     &             -deptgw)/s(nodepw(res_no,is,1))+ells(nodepw(res_no,is
     &             ,1))
                ENDIF
                DO ir = 1,noresincell(nodepw(res_no,is,1))
                   IF (iwmads(res_no,is) .EQ. ires_index_in_cell
     &             (nodepw(res_no,is,1),ir)) THEN
                     tail = stage_res(nodepw(res_no,is,1),ir)
                   ENDIF
                ENDDO
             ELSE
                IF (iwmads(res_no,is).gt.ntotres) then
                   deptgw =(ells(nodepw(res_no,is,1))-h(nodepw(res_no,is
     &             ,1))) * s(nodepw(res_no,is,1))
                   IF (pond(nodepw(res_no,is,1)).gt. deptgw) THEN
                      tail = pond(nodepw(res_no,is,1)) - deptgw
     &                + ells(nodepw(res_no,is,1))
                   ELSE
                      tail = (pond(nodepw(res_no,is,1))-deptgw)
     &                /s(nodepw(res_no,is,1)) + ells(nodepw(res_no,is,1))
```

```
                 ENDIF
              ELSE
                DO ir = 1,noresincell(nodepw(res_no,is,1))
                   IF (iwmads(res_no,is) .EQ. ires_index_in_cell
     &             (nodepw(res_no,is,1),ir)) THEN
                     tail = stage_res(nodepw(res_no,is,1),ir)
                   ENDIF
                ENDDO
              ENDIF
           ENDIF
          ENDIF
       ELSE
         tail=chdep(ircnl(res_no,is,1))+hdc(ircnl(res_no,is,1),iseas)
         wcrsth_s=crstelev(res_no,is,2)
       END IF
!
       IF (inodopt(res_no,is,2).eq.1) THEN
          head_av = head
          DO ir = 1,noresincell(idnstrm(res_no,is))
             IF (res_no .EQ. ires_index_in_cell
     &       (idnstrm(res_no,is),ir)) THEN
                head_av = (prev_stage_res(idnstrm(res_no,is),ir)
     &          + head)/2.0
             ENDIF
          ENDDO
       ELSE
          head_av = head
       ENDIF
       IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(27))
     &  THEN
         IF (sfactor(res_no) .GE. sfactmin .or. ires_small_sim
     &   (res_no) .eq. 'NO ') THEN
           IF (head.ge.24.0) THEN
             capc=400.
           ELSE IF (head.ge.20.0) THEN
             capc=350.
           ELSE IF (head.ge.15.0) THEN
             capc=300.
           ELSE IF (head.ge.crstelev(res_no,is,2)) THEN
             capc=300.
           END IF
         ELSE
           IF (head.ge.24.0) THEN
             capc=400.
           ELSE IF (head.ge.15.0) THEN
             capc=350.
           ELSE IF (head.ge. 5.0) THEN
             capc=300.
           ELSE IF (head.ge.crstelev(res_no,is,2)) THEN
             capc=300.
           END IF
         ENDIF
       ELSE
         IF (type(res_no,is).eq.'GRAVITY') THEN
           IF (head_av.gt.wcrsth_s) THEN
             capc=weircff(res_no,is,2)*(head_av-wcrsth_s)**wrexp(res_no
     &        ,is,2)
```

```
              capc = amin1(capc,RMXCPO(res_no,is))
            ELSE
              capc=0.0
            END IF
          ELSE
            capc=weircff(res_no,is,2)
            wcrsth_s=crstelev(res_no,is,2)
          END IF
        END IF
c       IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(21))
c    &  print *,weircff(res_no,is,2),head_av,wcrsth_s,wrexp(res_no,is,2)
c    &  ,capc,'   CAPACWCS4S  ',ells(idnstrm(res_no,is))
c    &  ,pond(idnstrm(res_no,is)),ibsn(idnstrm(res_no,is))
c    &  ,prev_stage_res(idnstrm(res_no,is),1),head,sfactor(res_no)
        IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(20)
     &   .or.res_out_strname(res_no,is).eq.res_struc_name_spec_code(19))
     &    capac_wcs4n=capc
        IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(21))
     &   capac_wcs4s=capc
        IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(37))
     $   capac_wcs4w=capc
        IF (inodopt(res_no,is,2).ne.1) THEN
          avail_vol=(chdep(ioutcnl(res_no,is))-wcrsth_s)*
     &     cl(ioutcnl(res_no,is))*width(ioutcnl(res_no,is))+
     &     tovflc(ioutcnl(res_no,is),wcrsth_s,0,0.,0,1)+
     &     cgseep(ioutcnl(res_no,is),wcrsth_s,0,0.)
     &     +qu(ioutcnl(res_no,is))
          avail_vol=avail_vol/86400.
          avail_vol=amax1(avail_vol,0.0)
          cout=amin1(avail_vol,capc)
        ELSE
!
         IF (sfactor(res_no) .GE. sfactmin .or. ires_small_sim
     &   (res_no) .eq. 'NO ') THEN
          gwdpth=(ells(idnstrm(res_no,is))-h(idnstrm(res_no,is)))*
     &     s(idnstrm(res_no,is))
          IF (pond(idnstrm(res_no,is)).ge.gwdpth) THEN
            stage=pond(idnstrm(res_no,is))-gwdpth+ells(idnstrm(res_no,
     &       is))
          ELSE
            stage=(pond(idnstrm(res_no,is))-gwdpth)/s(idnstrm(res_no,is)
     &       )+ells(idnstrm(res_no,is))
          END IF
          IF (stage.gt.ells(idnstrm(res_no,is))) THEN
            IF (sfactor(res_no).le.1.0) THEN
              sgn=1
            ELSE
              sgn=-1
            END IF
            term1_for_adjust_area=(sumet_outside_res(idnstrm(res_no,is))
     &       +seep_loss_res(idnstrm(res_no,is))+
     &       daily_seep_loss_res(idnstrm(res_no,is))+
     &       daily_et_outside_res(idnstrm(res_no,is))-sumrain(idnstrm
     &       (res_no,is))/12.0-sum_chg_depth_levee_spg(idnstrm(res_no
     &       ,is)))*((1.0-sfactor(res_no))/sfactor(res_no))
            term2_for_adjust_area=
     &       (outf_fc_frm_res_to_cell(idnstrm(res_no,is))+
```

```
&       outf_seep_loc_to_res_cell(idnstrm(res_no,is))+
&       daily_outf_fc_frm_res_to_cell(idnstrm(res_no,is)))/
&       sfactor(res_no)
        IF (sgn*term1_for_adjust_area.lt.term2_for_adjust_area)
&       THEN
         adjust_dur_excess(idnstrm(res_no,is))=
&         adjust_for_area(idnstrm(res_no,is))+
&         adjust_dur_deficit(idnstrm(res_no,is))
        ELSE
         adjust_dur_deficit(idnstrm(res_no,is))=
&         adjust_dur_excess(idnstrm(res_no,is))
         adjust_dur_excess(idnstrm(res_no,is))=
&         adjust_dur_deficit(idnstrm(res_no,is))
        END IF
        wsdph=pond(idnstrm(res_no,is))/sfactor(res_no)+
&       term1_for_adjust_area-term2_for_adjust_area-
&       adjust_dur_excess(idnstrm(res_no,is))
        IF (sgn*term1_for_adjust_area.ge.term2_for_adjust_area)
&       THEN
         adjust_for_area(idnstrm(res_no,is))=term1_for_adjust_area-
&         term2_for_adjust_area+daily_excess_adjust(idnstrm(res_no,
&         is))
         IF (sfactor(res_no).le.1.0) THEN
           adjust_for_area(idnstrm(res_no,is))=
&           amax1(adjust_for_area(idnstrm(res_no,is)),0.0)
         END IF
         adjust_for_area(idnstrm(res_no,is))=sgn*amin1(sgn*
&         adjust_for_area(idnstrm(res_no,is)),sgn*(wsdph-
&         pond(idnstrm(res_no,is))))
         wsdph=wsdph-adjust_for_area(idnstrm(res_no,is))
        END IF
        avail_vol=(ells(idnstrm(res_no,is))+wsdph-
&       stg_min_wsply(res_no,js))*gdar*sfactor(res_no)/86400.
      ELSE
        avail_vol=(ells(idnstrm(res_no,is))-stg_min_wsply(res_no,js)
&       )*gdar/86400.*sfactor(res_no)-(ells(idnstrm(res_no,is))-
&       stage)*s(idnstrm(res_no,is))*gdar/86400.
        wsdph=0.0
      END IF
     ELSE

       node_iresloc = idnstrm(res_no,is)
       DO ir = 1,noresincell(node_iresloc)
         IF (res_no .EQ. ires_index_in_cell
&       (node_iresloc,ir)) THEN
            IF (stg_min_wsply(res_no,js) .ge.ells_in_res(idnstrm(
&           res_no,is),ir)) THEN
               avail_vol=(head-stg_min_wsply(res_no,js))*gdar
&              *sfactor(res_no)/86400.
            ELSE
              avail_vol=(head-ells_in_res(idnstrm(res_no,is),ir))
&             *gdar*sfactor(res_no)/86400. + (ells_in_res
&             (idnstrm(res_no,is),ir)-stg_min_wsply(res_no,js))
&             *s(idnstrm(res_no,is))*gdar*sfactor(res_no)/86400.
            ENDIF
         ENDIF
       ENDDO
```

```
         ENDIF
         avail_vol=amax1(avail_vol,0.0)
         cout=amin1(avail_vol,capc)
        END IF
!
!  route water supply discharge(or seepage) to appropriate destination
!
        IF (res_out_type(res_no,is).eq.'SEEPG') THEN
          outflow_to_ds_cnl=cout
          IF (inodopt(res_no,is,3).ne.1) THEN
            DO kk=1,ndspts(res_no,is)
              dqu(ircnl(res_no,is,kk))=dqu(ircnl(res_no,is,kk))-
     &         amin1(outflow_to_ds_cnl,dqu(ircnl(res_no,is,kk)))
              icnl_dns=ircnl(res_no,is,kk)
   10         icnl_dns=iup_canal_ws(icnl_dns)
              IF (icnl_dns.le.nch) THEN
                dqu(icnl_dns)=dqu(icnl_dns)-outflow_to_ds_cnl
                dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
                GO TO 10
              END IF
              IF (iserv_area_loc(ircnl(res_no,is,kk)).ne.0) THEN
                rloc_excess_for_ws(iserv_area_loc(ircnl(res_no,is,kk)))=
     &           rloc_excess_for_ws(iserv_area_loc(ircnl(res_no,is,kk)))
     &           +outflow_to_ds_cnl*86400.
              END IF
              qu(ircnl(res_no,is,kk))=qu(ircnl(res_no,is,kk))+
     &         outflow_to_ds_cnl*86400.
              upstream_inflow(ircnl(res_no,is,kk))=
     &         upstream_inflow(ircnl(res_no,is,kk))+outflow_to_ds_cnl*
     &         86400.
            END DO
          ELSE
!             if (INODOPT(res_no,IS,2) .EQ. 1)
!    $        outf_seep_loc_to_res_cell(IDNSTRM(res_no,IS)) = 0.0
            DO kk=1,ndspts(res_no,is)
              pond(nodepw(res_no,is,kk))=pond(nodepw(res_no,is,kk))+
     &         cout*86400./(ndspts(res_no,is)*gdar*(1.0
     &         -carfact(nodepw(res_no,is,kk))-total_small_afact
     &         (nodepw(res_no,is,kk))))
              IF (nodepw(res_no,is,kk).eq.idnstrm(res_no,is)) THEN
                outf_seep_loc_to_res_cell(idnstrm(res_no,is))=
     &           outf_seep_loc_to_res_cell(idnstrm(res_no,is))+cout*
     &           86400./(ndspts(res_no,is)*gdar)
                daily_outf_seep_loc_to_res_cell(idnstrm(res_no,is))=
     &             cout*86400./(ndspts(res_no,is)*gdar)
              END IF
            END DO
          END IF
          outflow=cout
        ELSE IF (idshg_opt(res_no,js).eq.0) THEN
          IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(28)
     &     ) THEN
            outflow=amin1(cout,dqu(ic18))
            qu(ircnl(res_no,is,1))=qu(ircnl(res_no,is,1))+outflow*
     &       86400.
            upstream_inflow(ircnl(res_no,is,1))=
     &       upstream_inflow(ircnl(res_no,is,1))+outflow*86400.
```

```
          ELSE IF (res_out_strname(res_no,is)
     &     .eq.res_struc_name_spec_code(15)) THEN
            ds_dmnd=ws_down_sta2
            outflow=amin1(cout,ds_dmnd)
            outflow_from_sta2_for_ws=outflow
            IF (nint(ds_dmnd).gt.0) THEN
              ratio_del=outflow/ds_dmnd
            ELSE
              ratio_del=0.0
            END IF
            wsndes_rev=ratio_rem_sta2*(1.0-ratio_del)*wsndes
            wsndese_rev=ratio_rem_sta2*(1.0-ratio_del)*wsndese
            qu(ircnl(res_no,is,1))=qu(ircnl(res_no,is,1))+outflow*
     &       86400.
            upstream_inflow(ircnl(res_no,is,1))=
     &       upstream_inflow(ircnl(res_no,is,1))+outflow*86400.
          ELSE IF (res_out_strname(res_no,is)
     &     .eq.res_struc_name_spec_code(29)) THEN
            dsdmnd=dqu(is22up)+dqu(is25bup)
            outflow=amin1(cout,dsdmnd)
            IF (nint(dsdmnd).gt.0) THEN
              ratio_del=outflow/dsdmnd
            ELSE
              ratio_del=0.0
            END IF
            qu(is22up)=qu(is22up)+ratio_del*dqu(is22up)*86400.
            qu(is25bup)=qu(is25bup)+ratio_del*dqu(is25bup)*86400.
            upstream_inflow(is22up)=upstream_inflow(is22up)+ratio_del*
     &       dqu(is22up)*86400.
            upstream_inflow(is25bup)=upstream_inflow(is25bup)+ratio_del*
     &       dqu(is25bup)*86400.
            dqu(is22up)=(1.0-ratio_del)*dqu(is22up)
            dqu(is25bup)=(1.0-ratio_del)*dqu(is25bup)
            dqu(ircnl(res_no,is,1))=dqu(ircnl(res_no,is,1))-outflow
            icnl_dns=ircnl(res_no,is,1)
   20       icnl_dns=iup_canal_ws(icnl_dns)
            IF (icnl_dns.le.nch) THEN
              dqu(icnl_dns)=dqu(icnl_dns)-outflow
              dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
              GO TO 20
            END IF
C
C  SITE1 WS  (WSC1)
C
          ELSE IF (res_out_strname(res_no,is).eq.
     &     res_struc_name_spec_code(37)) THEN
            dsdmnd1lwdd = wsnde1
            dsdmnd2lwdd = wsnde2+wsnde3
            outflow1 = min(dsdmnd1lwdd,capaccs17w-kflo(iflcs17w))
            outflow2 = min(dsdmnd2lwdd,capaccs17e-kflo(iflcs17e))
            if (nint(dsdmnd2lwdd) .gt. 0) then
              ratio_del2 = outflow2/dsdmnd2lwdd
            else
              ratio_del2 = 0.0
            endif
            pot_deliv = outflow1+outflow2+dqu(ircnl(res_no,is,1))
            outflow = min(cout,pot_deliv)
```

```
        IF (nint(pot_deliv).gt.0) THEN
          ratio_del=outflow/pot_deliv
        ELSE
          ratio_del=0.0
        END IF
        qu(ie2)=qu(ie2)+ratio_del*ratio_del2*wsnde2*86400.
        qu(ie3)=qu(ie3)+ratio_del*ratio_del2*wsnde3*86400.
        kflo(iflcs17w)=kflo(iflcs17w)+ratio_del*outflow1
        kflo(iflcs17e)=kflo(iflcs17e)+ratio_del*outflow2
        wsnde1=wsnde1-ratio_del*outflow1
        wsnde2=(1.0-ratio_del*ratio_del2)*wsnde2
        wsnde3=(1.0-ratio_del*ratio_del2)*wsnde3
        qu(ircnl(res_no,is,1))=qu(ircnl(res_no,is,1))+outflow
&       *86400.
        dqu(ircnl(res_no,is,1)) = dqu(ircnl(res_no,is,1))
&          *(1.0-ratio_del)
      ELSE IF (res_out_strname(res_no,is).eq.'ACMRWS') THEN
        dsdmnd=wsnde1+wsnde2
        av_outf=amax1(capc-avail_vol_locfc,0.0)
        outflow=amin1(av_outf,avail_vol,dsdmnd)
        IF (nint(dsdmnd).gt.0) THEN
          ratio_del=outflow/dsdmnd
        ELSE
          ratio_del=0.0
        END IF
        qu(ie1)=qu(ie1)+ratio_del*wsnde1*86400.
        qu(ie2)=qu(ie2)+ratio_del*wsnde2*86400.
        wsnde1=(1.0-ratio_del)*wsnde1
        wsnde2=(1.0-ratio_del)*wsnde2
        ws_from_acmres_lwdd=outflow
        call match ('CPBRES',1,6,resname,ntotres,6,icpbres_no)
        rem_cap_to_cpbres=amax1(capc-outflow,0.0)
        if (mean_ponding_depth(res_no) .gt. 4.0) then
          avail_vol_to_cpbres=(mean_ponding_depth(res_no)
&            - 4.0)*res_area(res_no)/1.9835
          outflow_to_cpbres=amin1(rem_cap_to_cpbres
&         ,avail_vol_to_cpbres,volwtr(icpbres_no))
        else
          outflow_to_cpbres=0.0
        endif
        IF ( sfactor(icpbres_no) .GT. sfactmin
&       .OR. ires_small_sim(icpbres_no) .EQ. 'NO ') THEN
          pond(iupstrm(icpbres_no,1,1))=pond(iupstrm(icpbres_no
&         ,1,1))+outflow_to_cpbres*86400./gdar
        ELSE
            IF (inodopt(res_no,is,3).eq.1) THEN
               node_icpbres = iupstrm(icpbres_no,1,1)
               DO ir = 1,noresincell(node_icpbres)
                IF (icpbres_no .EQ. ires_index_in_cell
&               (node_icpbres,ir)) THEN
                  depth_to_wt = amax1(ells_in_res(node_icpbres,ir)
&                 - stage_res(node_icpbres,ir),0.0)*s(node_icpbres)
                  stage_res(node_icpbres,ir) = stage_res
&                 (node_icpbres,ir) + amin1(outflow_to_cpbres*86400.
&                 /(ndspts(res_no,is)*sfactor(icpbres_no)*gdar)
&                 ,depth_to_wt)/s(node_icpbres) + amax1
&                 (outflow_to_cpbres*86400./(ndspts(res_no,is)
```

```
     &                *sfactor(icpbres_no)*gdar) - depth_to_wt,0.0)
                      pond_in_res(node_icpbres,ir) = amax1(stage_res
     &                (node_icpbres,ir) - ells_in_res(node_icpbres
     &                ,ir),0.0)
                      res_struc_inflw(icpbres_no) =res_struc_inflw
     &                (icpbres_no) + outflow_to_cpbres*1.9835
                    ENDIF
                   ENDDO
                 ENDIF
               ENDIF

              outflow = outflow_to_cpbres + outflow
              ws_from_acmres_cpbres = outflow_to_cpbres
            ELSE IF (res_out_strname(res_no,is)
     &       .eq.res_struc_name_spec_code(30)) THEN
              dsdmnd=wsnde1+wsnde2+wsnde3
              outflow=amin1(cout,dsdmnd)
              IF (nint(dsdmnd).gt.0) THEN
                ratio_del=outflow/dsdmnd
              ELSE
                ratio_del=0.0
              END IF
              qu(ie1)=qu(ie1)+ratio_del*wsnde1
              qu(ie2)=qu(ie2)+ratio_del*wsnde2*86400.
              qu(ie3)=qu(ie3)+ratio_del*wsnde3*86400.
              wsnde1=(1.0-ratio_del)*wsnde1
              wsnde2=(1.0-ratio_del)*wsnde2
              wsnde3=(1.0-ratio_del)*wsnde3
            ELSE IF (res_out_strname(res_no,is)
     &       .eq.res_struc_name_spec_code(31)) THEN
              IF (dqu(ircnl(res_no,is,1)).gt.0.0.or.nint(flow_s155)
     &         .lt. 100) THEN
                outflow_to_l8s=amin1(cout,dqu(ircnl(res_no,is,1)))
                dqu(ircnl(res_no,is,1))=dqu(ircnl(res_no,is,1))-
     &           outflow_to_l8s
                qu(ircnl(res_no,is,1))=qu(ircnl(res_no,is,1))+
     &           outflow_to_l8s*86400.
                upstream_inflow(ircnl(res_no,is,1))=
     &           upstream_inflow(ircnl(res_no,is,1))+outflow_to_l8s*
     &           86400.
                icnl_dns=ircnl(res_no,is,1)
   30           icnl_dns=iup_canal_ws(icnl_dns)
                IF (icnl_dns.le.nch) THEN
                  dqu(icnl_dns)=dqu(icnl_dns)-outflow_to_l8s
                  dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
                  GO TO 30
                END IF
                IF (nint(flow_s155).lt.100) THEN
                  cout_to_c51_frm_l8bprs=amin1(cout-outflow_to_l8s
     &            ,100.0-flow_s155)
c                 s5a3_so = s5a3_so + cout_to_c51_frm_l8bprs
c                 kflo(iflbel5) = kflo(iflbel5) +
c    &            cout_to_c51_frm_l8bprs
                  qu(ic51w)=qu(ic51w)+cout_to_c51_frm_l8bprs*86400.
                ELSE
                  cout_to_c51_frm_l8bprs=0.0
                END IF
```

```
              outflow=outflow_to_l8s+cout_to_c51_frm_l8bprs
            ELSE
              outflow=0.0
              cout_to_c51_frm_l8bprs=0.0
              outflow_to_l8s=0.0
            END IF
          ELSE IF (res_out_strname(res_no,is)
     &     .eq.res_struc_name_spec_code(36)) THEN
            DO kk=1,nstrctr(ic11w)
              struc_name_s13a(kk)=tmpname(ic11w,kk)
            END DO
            CALL match ('S13A  ', 1, 6, struc_name_s13a, nstrctr(ic11w),
     &        6, istr_s13a)
            IF (chdep(ic11w).gt.chdep(ic11)+hdc(ic11,iseas)) THEN
              capac_s13a=discoef(ic11w,istr_s13a,1,1)*(chdep(ic11w)-
     &         chdep(ic11)-hdc(ic11,iseas))**pwr(ic11w,istr_s13a,1,1)
            ELSE
              capac_s13a=0.0
            END IF
            outflow=amin1(dqu(ircnl(res_no,is,1)),cout,capac_s13a)
            outf_from_c11res_to_c11w=outflow
            dqu(ircnl(res_no,is,1))=dqu(ircnl(res_no,is,1))-outflow
            qu(ircnl(res_no,is,1))=qu(ircnl(res_no,is,1))+outflow*
     &       86400.
            upstream_inflow(ircnl(res_no,is,1))=
     &       upstream_inflow(ircnl(res_no,is,1))+outflow*86400.
            icnl_dns=ircnl(res_no,is,1)
   40       icnl_dns=iup_canal_ws(icnl_dns)
            IF (icnl_dns.le.nch) THEN
              dqu(icnl_dns)=dqu(icnl_dns)-outflow
              dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
              GO TO 40
            END IF
          ELSE
            IF (.not.lwslo(i_ds_target_indx(res_no,is))) THEN
              ds_dmnd=0.0
              DO kk=1,ndspts(res_no,is)
                ds_dmnd=dqu(ircnl(res_no,is,kk))+ds_dmnd
              END DO
              IF (nint(ds_dmnd).gt.0) THEN
                ratio=amin1(cout/ds_dmnd,1.0)
              ELSE
                ratio=1.0
              END IF
              outflow=ratio*ds_dmnd
              DO kk=1,ndspts(res_no,is)
                outflow_to_ds_cnl=ratio*dqu(ircnl(res_no,is,kk))
c               if (resname(res_no) .eq. 'SITE1 ') print *,
c    $           cnm(ircnl(res_no,is,kk)),dqu(ircnl(res_no,is,kk))
c    $           ,chdep(ircnl(res_no,is,kk)),'  SITEO'
                dqu(ircnl(res_no,is,kk))=dqu(ircnl(res_no,is,kk))-
     &           outflow_to_ds_cnl
                icnl_dns=ircnl(res_no,is,kk)
   50           icnl_dns=iup_canal_ws(icnl_dns)
                IF (icnl_dns.le.nch) THEN
                  dqu(icnl_dns)=dqu(icnl_dns)-outflow_to_ds_cnl
                  dqu(icnl_dns)=amax1(dqu(icnl_dns),0.0)
```

```
                GO TO 50
              END IF
              qu(ircnl(res_no,is,kk))=qu(ircnl(res_no,is,kk))+
     &         outflow_to_ds_cnl*86400.
              upstream_inflow(ircnl(res_no,is,kk))=
     &         upstream_inflow(ircnl(res_no,is,kk))+outflow_to_ds_cnl*
     &         86400.
              if(resname(res_no).eq.'LKBLTN')tot_flw_to_c9r_frm_nlkblt
     $         = tot_flw_to_c9r_frm_nlkblt + outflow_to_ds_cnl
            END DO
          ELSE
            outflow=0.0
          END IF
        END IF
      ELSE IF (idshg_opt(res_no,js).eq.1) THEN
        IF (iconv_use_opt_ws(irs).eq.1) THEN
          IF (ieaabsn.eq.2) THEN
            rmax_cap_ws=cnlcap_ws(innr)+cnlcap_ws(ihill)
          ELSE IF (ieaabsn.eq.1) THEN
            rmax_cap_ws=cnlcap_ws(imiami)
          ELSE
            rmax_cap_ws=cnlcap_ws(iwpb)
          END IF
        ELSE
          rmax_cap_ws=9999999.
        END IF
        IF (stagelo_beg.lt.stgref_asr_rec.or.iasr_opt_eaa(ieaabsn)
     &     .eq.0) THEN
          outflow=amin1(cout,pot_del_to_eaa_basin
     &      (ieaabsn,indx_lok_del),rmax_cap_ws)
          IF (resname(res_no).eq.name_res_spec_code(9)
     &      .or.resname(res_no).eq.name_res_spec_code(10)) THEN
            IF (res_out_strname(res_no,is)
     &        .eq.res_struc_name_spec_code(24)
     &        .or.res_out_strname(res_no,is)
     &        .eq.res_struc_name_spec_code(25)) THEN
              ntarg=0
              DO itarg=1,no_targ_loc(is8bsn-6)
                isa=itarg_area(itarg,is8bsn-6)
                IF (ngrid_cells_daily(isa).gt.0) THEN
                  stage_import_wca(isa)=0.0
                  avg_sim_wca_stg(isa)=0.0
                  DO i=1,ngrid_cells_daily(isa)
                    stage_import_wca(isa)=stage_import_wca(isa)+
     &                daily_stg_import_wca(ngage_index_env(isa,i,1))/
     &                ngrid_cells_daily(isa)
!
!  compute simulated stages
!
                    depthgw=(ells(node_import_daily(isa,i))-
     &                h(node_import_daily(isa,i)))*
     &                s(node_import_daily(isa,i))
                    IF (pond(node_import_daily(isa,i)).gt.depthgw)
     &                THEN
                      wca_nodal_stage=pond(node_import_daily(isa,i))-
     &                  depthgw+ells(node_import_daily(isa,i))
                    ELSE
```

```
                  wca_nodal_stage=(pond(node_import_daily(isa,i))-
&                  depthgw)/s(node_import_daily(isa,i))+
&                  ells(node_import_daily(isa,i))
                END IF
                avg_sim_wca_stg(isa)=avg_sim_wca_stg(isa)+
&                wca_nodal_stage/ngrid_cells_daily(isa)
              END DO
              IF (avg_sim_wca_stg(isa).lt.stage_import_wca(isa)+
&              offset(isa,2)) ntarg=ntarg+1
            END IF
          END DO
          IF (ntarg.gt.0) outflow=0.0
        END IF
      END IF
      IF (ieaabsn.eq.2.and.agp(ieaabsn).gt.0.0) THEN
        IF (alocnnr_potdel(indx_lok_del).gt.0.0) THEN
          alocnnr_potdel(indx_lok_del)=alocnnr_potdel(indx_lok_del
&         )-(outflow/pot_del_to_eaa_basin(ieaabsn,indx_lok_del))
&         *alocnnr_potdel(indx_lok_del)
        END IF
      END IF
      IF (iconv_use_opt_ws(irs).eq.1) THEN
        IF (ieaabsn.eq.2) THEN
          IF (agp(ieaabsn).gt.0.0) THEN
            IF (nint(rmax_cap_ws).gt.0) THEN
              cnlcap_ws(innr)=cnlcap_ws(innr)-(cnlcap_ws(innr)/
&              rmax_cap_ws)*outflow
              cnlcap_ws(ihill)=cnlcap_ws(ihill)-(cnlcap_ws(ihill)/
&              rmax_cap_ws)*outflow
            END IF
          END IF
        ELSE IF (ieaabsn.eq.1) THEN
          cnlcap_ws(imiami)=cnlcap_ws(imiami)-outflow
        ELSE
          cnlcap_ws(iwpb)=cnlcap_ws(iwpb)-outflow
        END IF
      END IF
      IF (ieaabsn.gt.3) agp(ieaabsn)=agp(ieaabsn)
&     - outflow*86400.
      pot_del_to_eaa_basin(ieaabsn,indx_lok_del) =
&     pot_del_to_eaa_basin(ieaabsn,indx_lok_del) - outflow
      IF (ieaabsn.eq.il8bsn-6) dmndagl8=agp(ieaabsn)/43560.
    ELSE
      IF (ieaabsn .le. 3) THEN
        iday_of_week = idofw(month,iday,ifyr+iyear-1)
        outflow = 0.0
        DO IW = 1,n_days_week_del_eaa(indx_lok_del)
           IF (iday_of_week .EQ. idays_wk_del_eaa_ssm_indx
+          (iw,indx_lok_del)) THEN
             eaa_dmnd_bsin=min(iagflo(iag(ieaabsn+6),1),0)
             eaa_dmnd_bsin=abs(eaa_dmnd_bsin)
             outflow=amin1(cout,eaa_dmnd_bsin,rmax_cap_ws)
             IF (iagflo(iag(ieaabsn+6),1).lt.0) THEN
                iagflo(iag(ieaabsn+6),1)=iagflo(iag(ieaabsn+6),1)+
&               ifix(outflow)
             END IF
           END IF
```

```
              ENDDO
            ELSE
              eaa_dmnd_bsin=min(iagflo(iag(ieaabsn+6),1),0)
              eaa_dmnd_bsin=abs(eaa_dmnd_bsin)
              outflow = amin1(cout,eaa_dmnd_bsin,rmax_cap_ws)
              IF (iagflo(iag(ieaabsn+6),1).lt.0) THEN
                  iagflo(iag(ieaabsn+6),1)=iagflo(iag(ieaabsn+6),1)+
     &            ifix(outflow)
              END IF
            ENDIF
          ENDIF
          res_outflow_to_basin(irs,js)=outflow
        ELSE IF (idshg_opt(res_no,js).eq.3) THEN
          IF (resname(iwmads(res_no,is)).eq. name_res_spec_code(1)) THEN
c         IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(19)
c    &     .or.res_out_strname(res_no,is)
c    &     .eq.res_struc_name_spec_code(20).or.res_out_strname(res_no,
c    &     is).eq.res_struc_name_spec_code(21)
c    &     .or.res_out_strname(res_no,is)
c    &     .eq.res_struc_name_spec_code(22).or.res_out_strname(res_no,
c    &     is).eq.res_struc_name_spec_code(23)) THEN
            if (iopt_env_eaar_thru_g372 .eq. 1) then
              rem_capac_to_sta=amax1(g372cap-runoff_to_sta,0.0)-
     &        res_outflow_env
            else
              rem_capac_to_sta=999999.
            endif
!
!  determine if criteria are met for environmental ws releases
!   from RESERVOIR to WCA-3A
!
            ntarg=0
            IF (sta34_outf_flex_eaar_opt.eq.'NO ') THEN
              IF (resname(res_no).eq.name_res_spec_code(10)
     &         .or.resname(res_no).eq.name_res_spec_code(11)
     &         .or.resname(res_no).eq.name_res_spec_code(9)) THEN
                DO itarg=1,no_targ_loc(is8bsn-6)
                  IF (lwslo(itarg_area(itarg,is8bsn-6))) THEN
                    ntarg=ntarg+1
                  END IF
                END DO
              ELSE
                DO itarg=1,no_targ_loc(is8bsn-6)
                  isa=itarg_area(itarg,is8bsn-6)
                  IF (avg_sim_wca_stg(isa).lt.stage_import_wca(isa)+
     &             offset(isa,1)) ntarg=ntarg+1
                END DO
              END IF
            ELSE
              DO itarg=1,no_targ_loc(is8bsn-6)
                IF (lwslo_eaares(itarg_area(itarg,is8bsn-6))) THEN
                  ntarg=ntarg+1
                END IF
              END DO
            END IF
            IF (ntarg.gt.0) THEN
              IF (res_out_strname(res_no,is)
```

```
     &         .eq.res_struc_name_spec_code(22)
     &         .or.res_out_strname(res_no,is)
     &         .eq.res_struc_name_spec_code(23)) THEN
               IF (res_out_strname(res_no,is)
     &           .eq.res_struc_name_spec_code(22)) capac_wcs4=
     &           capac_wcs4n
               IF (res_out_strname(res_no,is)
     &           .eq.res_struc_name_spec_code(23)) capac_wcs4=
     &           capac_wcs4s
               IF (nint(capac_wcs4).eq.0) THEN
                 outflow=amin1(cout,volwtr(iwmads(res_no,is)),
     &            rem_capac_to_sta)
               ELSE
                 outflow=0.0
               END IF
c              print *,res_out_strname(res_no,is),capac_wcs4,
c    &          cout,volwtr(iwmads(res_no,is)),rem_capac_to_sta
c    &          ,'  KLKL  ',capac_wcs4n,capac_wcs4s,outflow
             ELSE
               outflow=amin1(cout,volwtr(iwmads(res_no,is)),
     &           rem_capac_to_sta)
             END IF
           ELSE
             outflow=0.0
           END IF
           res_outflow_env=res_outflow_env+outflow
           volwtr(iwmads(res_no,is))=volwtr(iwmads(res_no,is))-outflow
           if(iopt_env_eaar_thru_g372 .eq. 1) then
            env_ws_from_res_thru_g372=outflow+env_ws_from_res_thru_g372
           endif
         ELSEIF (resname(iwmads(res_no,is)).eq. name_res_spec_code(4))
     &   THEN
            rem_capac_to_sta=amax1(capc-runoff_to_sta,0.0)
            IF (res_out_strname(res_no,is)
     &         .eq.res_struc_name_spec_code(38)) THEN
               capac_wcs4 = capac_wcs4w
               IF (nint(capac_wcs4).eq.0) THEN
                 IF (LWSLO(i_ds_target_indx(res_no,is))) THEN
                   outflow = amin1(cout,volwtr(iwmads(res_no,is)),
     &             rem_capac_to_sta)
                 ELSE
                   outflow = 0.0
                 ENDIF
               ELSE
                 outflow = 0.0
               ENDIF
c              print *,capac_wcs4,LWSLO(i_ds_target_indx(res_no,is))
c    &          ,outflow,cout,volwtr(iwmads(res_no,is))
c    &          ,resname(iwmads(res_no,is)),'  WCS4W  ',capc
c    &          ,runoff_to_sta
            ELSE
              IF (LWSLO(i_ds_target_indx(res_no,is))) THEN
                outflow = amin1(cout,volwtr(iwmads(res_no,is)),
     &          rem_capac_to_sta)
              ELSE
                outflow = 0.0
              ENDIF
```

```
            ENDIF
            QINFLW(indxsta(10),1) = OUTFLOW +QINFLW(indxsta(10),1)
            VOLWTR(INDXSTA(10)) = VOLWTR(INDXSTA(10)) - OUTFLOW
         ELSEIF (resname(iwmads(res_no,is)) .eq. 'NORES ') THEN
            IF (LWSLO(i_ds_target_indx(res_no,is))) THEN
               outflow = cout
            ELSE
               outflow = 0.0
            ENDIF
         END IF
         total_env_ws_from_eaares=outflow+total_env_ws_from_eaares
!        print *,outflow,i_ds_target_indx(res_no,is),ntarg
!    $   ,LWSLO(i_ds_target_indx(res_no,is)),g372cap
!    $   ,runoff_to_sta,rem_capac_to_sta,iwmads(res_no,is)
!    $   ,volwtr(iwmads(res_no,is)),capc,'  XXXXX'
         IF (resname(iwmads(res_no,is)).ne. name_res_spec_code(4))
     $    THEN
           DO j=1,ndspts(res_no,is)
             IF (inodopt(res_no,is,3).eq.1) THEN
               pond(nodepw(res_no,is,j))=pond(nodepw(res_no,is,j))+
     &          outflow*86400./(ndspts(res_no,is)*gdar*(1.0 -
     &          carfact(nodepw(res_no,is,j))-total_small_afact
     &          (nodepw(res_no,is,j))))
             ELSE
               qu(ircnl(res_no,is,j))=qu(ircnl(res_no,is,j))+outflow*
     &         86400./ndspts(res_no,is)
               upstream_inflow(ircnl(res_no,is,j))=
     &          upstream_inflow(ircnl(res_no,is,j))+outflow*86400./
     &          ndspts(res_no,is)
             END IF
           END DO
         ENDIF
       ELSE
         IF (isem_flg.eq.1) THEN
c           IF (res_out_strname(res_no,is)
c     &      .eq.res_struc_name_spec_code(32)) THEN
c             capac_sugh=capc
c             dmnd_sugh_orig=amax1(dmdsugh,0.0)
c             outflow=amin1(cout,dmnd_sugh_orig/1.9835,capc)
c             dmnd_sugh=dmnd_sugh_orig-outflow*1.9835
c             iagflo(iag(isugbsn),1)=-max(ifix(dmnd_sugh_orig/1.9835-
c     &        capc),0)+iagflo(iag(isugbsn),1)
c           ELSE
              outflow=0.0
c          END IF
         ELSE
           outflow=0.0
         END IF
       END IF
!
       IF (idshg_opt(res_no,js).ne.1) THEN
         IF (inodopt(res_no,is,2).ne.1) THEN
           qu(ioutcnl(res_no,is))=qu(ioutcnl(res_no,is))-outflow*
     &      86400.
         ELSE
           IF (sfactor(res_no) .GE. sfactmin .or. ires_small_sim
     &     (res_no) .eq. 'NO ') THEN
```

```
              pond(idnstrm(res_no,is))=pond(idnstrm(res_no,is))-outflow*
     &        86400./gdar
            ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
            DO ir = 1,noresincell(idnstrm(res_no,is))
             IF (res_no .eq. ires_index_in_cell(idnstrm(res_no,is)
     &       ,ir)) THEN
               IF (outflow .gt. 0.000001) THEN
                  stage_res(idnstrm(res_no,is),ir)=stage_res
     &            (idnstrm(res_no,is),ir)-amin1(outflow*86400.
     &            /(sfactor(res_no)*gdar),pond_in_res(idnstrm(res_no,is)
     &            ,ir)) - amax1(outflow*86400./(sfactor(res_no)
     &            *gdar) - pond_in_res(idnstrm(res_no,is),ir),0.0)
     &            / s(idnstrm(res_no,is))
                  pond_in_res(idnstrm(res_no,is),ir)=pond_in_res(idnstrm
     &            (res_no,is),ir) - amin1(outflow*86400./(sfactor
     &            (res_no)*gdar),pond_in_res(idnstrm(res_no,is),ir))
               ELSE
                  sgn = -1.0
                  depth_to_wt = amax1(ells_in_res(idnstrm(res_no,is),ir)
     &            - stage_res(idnstrm(res_no,is),ir),0.0)
     &            *s(idnstrm(res_no,is))
                  stage_res(idnstrm(res_no,is),ir)=stage_res
     &            (idnstrm(res_no,is),ir)+ amin1(sgn*outflow*86400.
     &            /(sfactor(res_no)*gdar),depth_to_wt)/s(idnstrm(res_no
     &            ,is))+amax1(sgn*outflow*86400./(sfactor(res_no)*gdar)
     &            - depth_to_wt,0.0)
                  pond_in_res(idnstrm(res_no,is),ir) = amax1(stage_res
     &            (idnstrm(res_no,is),ir) - ells_in_res(idnstrm(res_no
     &            ,is),ir),0.0)
               ENDIF
             ENDIF
            ENDDO
           ENDIF
          ENDIF
          JSTR = JSTR + 1
          STRFLW(JSTR,IDSS) = outflow
        ELSE
          IF (sfactor(res_no) .GE. sfactmin .or. ires_small_sim
     &    (res_no) .eq. 'NO ') THEN
            pond(idnstrm(res_no,is))=pond(idnstrm(res_no,is))-outflow*
     &       86400./gdar
          ELSE
c  reservoir is small ,treated a separate entity in grid domain
c
            DO ir = 1,noresincell(idnstrm(res_no,is))
             IF (res_no .eq. ires_index_in_cell(idnstrm(res_no,is)
     &       ,ir)) THEN
               IF (outflow .gt. 0.000001) THEN
                  stage_res(idnstrm(res_no,is),ir)=stage_res
     &            (idnstrm(res_no,is),ir)-amin1(outflow*86400.
     &            /(sfactor(res_no)*gdar),pond_in_res(idnstrm(res_no,is)
     &            ,ir)) - amax1(outflow*86400./(sfactor(res_no)
     &            *gdar) - pond_in_res(idnstrm(res_no,is),ir),0.0)
     &            / s(idnstrm(res_no,is))
                  pond_in_res(idnstrm(res_no,is),ir)=pond_in_res(idnstrm
```

```
     &            (res_no,is),ir) - amin1(outflow*86400./(sfactor
     &            (res_no)*gdar),pond_in_res(idnstrm(res_no,is),ir))
                ELSE
                   sgn = -1.0
                   depth_to_wt = amax1(ells_in_res(idnstrm(res_no,is),ir)
     &            - stage_res(idnstrm(res_no,is),ir),0.0)
     &            *s(idnstrm(res_no,is))
                   stage_res(idnstrm(res_no,is),ir)=stage_res
     &            (idnstrm(res_no,is),ir)+ amin1(sgn*outflow*86400.
     &            /(sfactor(res_no)*gdar),depth_to_wt)/s(idnstrm(res_no
     &            ,is))+amax1(sgn*outflow*86400./(sfactor(res_no)*gdar)
     &            - depth_to_wt,0.0)
                   pond_in_res(idnstrm(res_no,is),ir) = amax1(stage_res
     &            (idnstrm(res_no,is),ir) - ells_in_res(idnstrm(res_no
     &            ,is),ir),0.0)
                ENDIF
             ENDIF
            ENDDO
          ENDIF
        END IF
      ELSE
        IF (res_out_strname(res_no,is).eq.res_struc_name_spec_code(16))
     &   THEN
          iwca=idn_wca_indx_env(is8bsn-6)
          capc=weircff(res_no,is,2)
          capac_semws=capc
          avail_vol=outflow_from_roten_to_west
          outflow=amin1(avail_vol,sem_cyp_dmnd(jday)/1.9835,capc)
!         print *,sem_cyp_dmnd(jday)/1.9835,capc,'  SEMCYP'
!    $    ,outflow
          sem_cyp_dmnd(jday)=sem_cyp_dmnd(jday)-outflow*1.9835
          outflow_from_roten_to_wca=outflow_from_roten_to_west-outflow
          outflow_from_roten_to_semcyp=outflow
          jstr=jstr+1
          strflw(jstr,idss)=outflow
          IF (wcaenv(iwca).eq.'TRUE '.or.isem_flg.eq.1) THEN
            if (issm_cutb_opt_bcyp .eq. 1) then
               rem_dmnd_sem_cyp=sem_cyp_dmnd(jday)*pctmet/1.9835
            else
               rem_dmnd_sem_cyp=sem_cyp_dmnd(jday)/1.9835
            endif
            rem_cap_new_pump=capc_new_pump_to_west_frm_roten-
     &       outflow_from_roten_to_west
            rem_cap_for_sply_sem=capac_semws-(sem_dmnd_orig/1.9835-
     &       sem_cyp_dmnd(jday)/1.9835)
            rem_cap_for_sply_sem=amax1(rem_cap_for_sply_sem,0.0)
            rem_cap_for_sply_sem=amin1(rem_cap_for_sply_sem,
     &       rem_dmnd_sem_cyp,rem_cap_new_pump)
            rem_cap_new_pump=rem_cap_new_pump-rem_cap_for_sply_sem
            IF (wcaenv(iwca).eq.'TRUE ') THEN
              flow_thru_s140_restor=amin1(outflow_from_roten_to_wca,
     &         avail_cap_s140)
            ELSE
              flow_thru_s140_restor=0.0
            END IF
            IF (wcaenv(iwca).eq.'TRUE ') THEN
              IF (.not.lwslo(inwcwca3a).or.(.not.lwslo(inwcwca3a)
```

```
     &         .and..not.lwslo_eaares(inwcwca3a).and.((stg_eaa_res_2a
     &         .ge.stg_eaar_min_2a.and.ieaarsn_no.gt.0)
     &         .or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.ieaarss_no
     &         .gt.0)).and.sta34_outf_flex_eaar_opt.eq.'YES')) THEN
                 qu(is140up)=qu(is140up)+flow_thru_s140_restor*86400.
                 outflow_from_roten_to_wca=outflow_from_roten_to_wca-
     &            flow_thru_s140_restor
                 avail_cap_s140=avail_cap_s140-flow_thru_s140_restor
c                print *,qu(is140up)/86400.,flow_thru_s140_restor
c    &            ,avail_cap_s140,'  S140ROT'
                 roten_out_to_s140=flow_thru_s140_restor
                 IF (.not.lwslo(iwca3a_3a2).or.(.not.lwslo(iwca3a_3a2)
     &           .and..not.lwslo_eaares(iwca3a_3a2).and.
     &           ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no.gt.0)
     &           .or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.ieaarss_no
     &           .gt.0)).and.sta34_outf_flex_eaar_opt.eq.'YES')) THEN
                   holy_out_to_s140  = 0.0
                   holey_out_to_ds    = total_outf_from_holy
                   sta34_out_to_s140 = 0.0
                 ELSE
                   flow_thru_s140_holy=amin1(total_outf_from_holy,
     &              avail_cap_s140,rem_cap_new_pump)
                   qu(is140up)=qu(is140up)+flow_thru_s140_holy*86400.
                   avail_cap_s140=avail_cap_s140-flow_thru_s140_holy
c                  print *,qu(is140up)/86400.,flow_thru_s140_holy
c    &             ,avail_cap_s140,'  S140HLY'
                   rem_cap_new_pump=rem_cap_new_pump-flow_thru_s140_holy
                   total_outf_from_holy=total_outf_from_holy-
     &              flow_thru_s140_holy
                   holy_out_to_s140=flow_thru_s140_holy
                   holey_out_to_ds=total_outf_from_holy
                   IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
     &             .and..not.lwslo_eaares(inewca3a).and.
     &             ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &             .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &             ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &             'YES')) THEN
                     total_avail_flow_sta34=dischg_to_s7_frm_sta34+
     &                pot_exc_outf_frm_sta34_to_nca3a+frac_thru_s8_sta34*
     &                dischg_to_s8_frm_sta34
                   ELSE
                     total_avail_flow_sta34=frac_thru_s8_sta34*
     &                dischg_to_s8_frm_sta34+dischg_to_s7_frm_sta34
                   END IF
                   sta34_out_to_s140=amin1(avail_cap_s140,
     &              total_avail_flow_sta34,rem_cap_new_pump)
                   rem_cap_new_pump=rem_cap_new_pump-sta34_out_to_s140
                   excess_flow=amax1(sta34_out_to_s140-
     &              frac_thru_s8_sta34*dischg_to_s8_frm_sta34,0.0)
                   IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
     &             .and..not.lwslo_eaares(inewca3a).and.
     &             ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &             .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &             ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt
     &             .eq.'YES')) THEN
                     total_flow_wca3a = frac_thru_s8_sta34
     $               * dischg_to_s8_frm_sta34
```

```
     $              + pot_exc_outf_frm_sta34_to_nca3a
                  ELSE
                    total_flow_wca3a = frac_thru_s8_sta34
     $              *dischg_to_s8_frm_sta34
                  END IF
                  excess_flow_nca3a=amax1(sta34_out_to_s140-
     &             total_flow_wca3a,0.0)
                  dischg_to_s8_frm_sta34=dischg_to_s8_frm_sta34-
     &             amin1(sta34_out_to_s140,frac_thru_s8_sta34*
     &             dischg_to_s8_frm_sta34)
                  avail_cap_s140=avail_cap_s140-sta34_out_to_s140
                  qu(is140up)=qu(is140up)+sta34_out_to_s140*86400.
c                 print *,qu(is140up)/86400.,sta34_out_to_s140
c    &              ,'  S140STA34'
                  IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
     &            .and..not.lwslo_eaares(inewca3a).and.
     &            ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &            .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &            ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &            'YES')) THEN
                    pot_exc_outf_frm_sta34_to_nca3a=
     &               pot_exc_outf_frm_sta34_to_nca3a-amin1(excess_flow,
     &               pot_exc_outf_frm_sta34_to_nca3a)
                  END IF
                  dischg_to_s7_frm_sta34=dischg_to_s7_frm_sta34-
     &             amin1(dischg_to_s7_frm_sta34,excess_flow_nca3a)
                END IF
                holey_out_to_nwc=0.0
                sta34_out_to_wca3anw=0.0
                total_outflow_to_wca=outflow_from_roten_to_wca
              ELSE
                roten_out_to_s140=0.0
                IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
     &          .and..not.lwslo_eaares(inewca3a).and.
     &          ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &          ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &          'YES')) THEN
                  total_avail_flow_sta34=dischg_to_s7_frm_sta34+
     &             pot_exc_outf_frm_sta34_to_nca3a+frac_thru_s8_sta34*
     &             dischg_to_s8_frm_sta34
                ELSE
                  total_avail_flow_sta34=frac_thru_s8_sta34*
     &             dischg_to_s8_frm_sta34+dischg_to_s7_frm_sta34
                END IF
                outflow_from_holy_west=amin1(total_outf_from_holy,
     &           rem_cap_new_pump)
                total_outf_from_holy=total_outf_from_holy-
     &           outflow_from_holy_west
                rem_cap_new_pump=rem_cap_new_pump-
     &           outflow_from_holy_west
                outflow_from_sta34=amin1(total_avail_flow_sta34,
     &           rem_cap_new_pump)
                IF (.not.lwslo(iwca3a_3a2).or.(.not.lwslo(iwca3a_3a2)
     &          .and..not.lwslo_eaares(iwca3a_3a2).and.
     &          ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
```

```
     &          ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &          'YES')) THEN
                  holy_out_to_s140=0.0
                  holey_out_to_nwc=outflow_from_holy_west
                  holey_out_to_ds=total_outf_from_holy
                  sta34_out_to_s140=0.0
                  sta34_out_to_wca3anw=outflow_from_sta34
                ELSE
                  holy_out_to_s140=amin1(0.5*outflow_from_holy_west,
     &             avail_cap_s140)
                  excess_flow_holy_s140=amax1(0.5*
     &             outflow_from_holy_west-avail_cap_s140,0.0)
                  avail_cap_s140=avail_cap_s140-holy_out_to_s140
                  qu(is140up)=qu(is140up)+holy_out_to_s140*86400.
c                 print *,qu(is140up)/86400.,holy_out_to_s140
c    &              ,avail_cap_s140,'   S140HLY'
                  holey_out_to_nwc=0.5*outflow_from_holy_west+
     &             excess_flow_holy_s140
                  holey_out_to_ds=total_outf_from_holy
                  outflow_from_sta34_s140=amin1(0.5*outflow_from_sta34,
     &             avail_cap_s140)
                  excess_flow_s140=0.5*outflow_from_sta34-
     &             avail_cap_s140
                  excess_flow_s140=amax1(excess_flow_s140,0.0)
                  sta34_out_to_s140=outflow_from_sta34_s140
                  sta34_out_to_wca3anw=0.5*outflow_from_sta34+
     &             excess_flow_s140
                  avail_cap_s140=avail_cap_s140-outflow_from_sta34_s140
                  qu(is140up)=qu(is140up)+outflow_from_sta34_s140*
     &             86400.
c                 print *,qu(is140up)/86400.,outflow_from_sta34_s140
c    &              ,avail_cap_s140,'  S140STA34'
                END IF
                total_outflow_to_wca=outflow_from_roten_to_wca+
     &           sta34_out_to_wca3anw
                excess_flow_west=amax1(outflow_from_sta34-
     &           frac_thru_s8_sta34*dischg_to_s8_frm_sta34,0.0)
                avail_flow_s8=amin1(outflow_from_sta34,
     &           frac_thru_s8_sta34*dischg_to_s8_frm_sta34)
                IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
     &          .and..not.lwslo_eaares(inewca3a).and.
     &          ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &          ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &          'YES')) THEN
                  total_flow_wca3a=frac_thru_s8_sta34*
     &             dischg_to_s8_frm_sta34+
     &             pot_exc_outf_frm_sta34_to_nca3a
                ELSE
                  total_flow_wca3a=frac_thru_s8_sta34*
     &             dischg_to_s8_frm_sta34
                END IF
                reduc_flow_nwca3a=amax1(outflow_from_sta34-
     &           total_flow_wca3a,0.0)
                dischg_to_s8_frm_sta34=dischg_to_s8_frm_sta34-
     &           avail_flow_s8
                IF (.not.lwslo(inewca3a).or.(.not.lwslo(inewca3a)
```

```
     &          .and..not.lwslo_eaares(inewca3a).and.
     &          ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &          ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &          'YES')) THEN
                  pot_exc_outf_frm_sta34_to_nca3a=
     &             pot_exc_outf_frm_sta34_to_nca3a-
     &             amin1(excess_flow_west,
     &             pot_exc_outf_frm_sta34_to_nca3a)
                END IF
                dischg_to_s7_frm_sta34=dischg_to_s7_frm_sta34-
     &           amin1(dischg_to_s7_frm_sta34,reduc_flow_nwca3a)
              END IF
              iroute_to_wca3a_frm_holy=1
            ELSE
              sta34_out_to_wca3anw=0.0
              roten_out_to_s140=0.0
              total_outflow_to_wca=outflow_from_roten_to_wca
              holy_out_to_s140=0.0
              holey_out_to_nwc=0.0
              holey_out_to_ds=total_outf_from_holy
              IF (hlyenv.eq.'TRUE ') THEN
                iroute_to_wca3a_frm_holy=1
              ELSE
                iroute_to_wca3a_frm_holy=0
              END IF
            END IF
          ELSE
            sta34_out_to_wca3anw=0.0
            roten_out_to_s140=0.0
            total_outflow_to_wca=outflow_from_roten_to_wca
            holy_out_to_s140=0.0
            holey_out_to_nwc=0.0
            holey_out_to_ds=total_outf_from_holy
            IF (hlyenv.eq.'TRUE ') THEN
              iroute_to_wca3a_frm_holy=1
            ELSE
              iroute_to_wca3a_frm_holy=0
            END IF
          END IF
c       print *,sta34_out_to_wca3anw,sta34_out_to_s140
c    $    ,pot_exc_outf_frm_sta34_to_nca3a
c    $    ,sta34_out_to_wca3ane,dischg_to_s8_frm_sta34,' STA34 '
c    $    ,LWSLO(inwcwca3a),LWSLO(iwca3a_3a2),LWSLO(inewca3a)
c    $    ,outflow_from_sta34,total_avail_flow_sta34
c    $    ,rem_cap_new_pump,total_outflow_from_sta34
c    $    ,dischg_to_s7_frm_sta34
c       print *,holey_out_to_ds,holey_out_to_nwc,holy_out_to_s140
c    $            ,'  HOLEY ',LWSLO(inwcwca3a),LWSLO(iwca3a_3a2)
c    $            ,total_outf_from_holy
          DO j=1,ndspts(res_no,is)
            IF (inodopt(res_no,is,3).eq.1) THEN
              pond(nodepw(res_no,is,j))=pond(nodepw(res_no,is,j))+
     &         total_outflow_to_wca*86400./(ndspts(res_no,is)*gdar*(1.0
     &         - carfact(nodepw(res_no,is,j))-total_small_afact
     &         (nodepw(res_no,is,j))))
            ELSE
```

```
              qu(ircnl(res_no,is,j))=qu(ircnl(res_no,is,j))+
     &         total_outflow_to_wca*86400./ndspts(res_no,is)
            END IF
          END DO
          IF (iroute_to_wca3a_frm_holy.eq.1) THEN
            IF (iroute_to_miamic_via_g200a.eq.0) THEN
!           print *,iroute_to_wca3a_frm_holy,holey_out_to_ds
!    $          ,holey_out_to_nwc,holy_out_to_s140,'  CCCCC'
              qu(idculv1)=qu(idculv1)+holey_out_to_ds*0.65*86400.
              qu(idculv2)=qu(idculv2)+holey_out_to_ds*0.35*86400.
            END IF
            pond(nodnwc3a)=pond(nodnwc3a)+holey_out_to_nwc*86400./(gdar
     $      *(1.0 - carfact(nodnwc3a)))
          END IF
!
!  route outflow to northeastern wca3a thru gaps from sta34
!  sta34 is reservoir #1
!
!        qoutf = pot_exc_outf_frm_sta34_to_nca3a
!        if (.not. LWSLO(inewca3a) .and.
!    $         opt_outflow_from_res_to_wca .ne. 'FLDC') then
!          sta34_out_to_wca3ane = qoutf
!        endif
          flow_to_newca3a=pot_exc_outf_frm_sta34_to_nca3a
          CALL match (name_res_spec_code(1), 1, 6, resname, ntotres+1,
     &      6, irsta34)
          DO ks=1,no_gravity_outlets_sta34
            is=igrav_outlet_no(ks)
            IF (res_out_strname(irsta34,is)
     &       .eq.res_struc_name_spec_code(1).or.res_out_strname(irsta34,
     &       is).eq.res_struc_name_spec_code(2)
     &       .or.res_out_strname(irsta34,is)
     &       .eq.res_struc_name_spec_code(3)) THEN
!           if (opt_outflow_from_res_to_wca .eq. 'FLDC' .or.
!    $        LWSLO(inewca3a)) then
!            do j = 1,NDSPTS(irsta34,IS)
!              IF (INODOPT(irsta34,IS,3) .EQ. 1) THEN
!                POND(NODEPW(irsta34,IS,J)) = POND(NODEPW(irsta34,IS,J))
!    $            + QOUTF*86400. /(no_gravity_outlets_sta34
!    $            *NDSPTS(irsta34,IS) * GDAR)
!              ELSE
!                QU(IRCNL(irsta34,IS,J)) = QU(IRCNL(irsta34,IS,J))
!    $           + QOUTF * 86400./(no_gravity_outlets_sta34
!    $           * NDSPTS(irsta34,IS))
!              ENDIF
!            enddo
              IF (((.not.lwslo(inewca3a).and..not.lwslo(iwca3a))
     &         .or.(.not.lwslo(inewca3a).and..not.lwslo_eaares(inewca3a)
     &         .and. ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no
     &          .gt.0).or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.
     &          ieaarss_no.gt.0)).and.sta34_outf_flex_eaar_opt.eq.
     &          'YES')).and.opt_outflow_from_res_to_wca .ne.'FLDC') THEN
                DO j=1,ndspts(irsta34,is)
                  IF (inodopt(irsta34,is,3).eq.1) THEN
                    pond(nodepw(irsta34,is,j))=pond(nodepw(irsta34,is,j)
     &               )+flow_to_newca3a*86400./(no_gravity_outlets_sta34*
     &               ndspts(irsta34,is)*gdar)
```

```
                  ELSE
                    qu(ircnl(irsta34,is,j))=qu(ircnl(irsta34,is,j))+
     &               flow_to_newca3a*86400./(no_gravity_outlets_sta34*
     &               ndspts(irsta34,is))
                  END IF
                END DO
                sta34_out_to_wca3ane=flow_to_newca3a
              END IF
            END IF
          END DO
        ELSE
          iwca=idn_wca_indx_env(is8bsn-6)
          avail_vol=outflow_from_sta6
          outflow=amin1(avail_vol,sem_cyp_dmnd(jday)/1.9835,capac_semws)
!         print *,avail_vol,outflow,capac_semws,sem_cyp_dmnd(jday)/1.9835
!    $      ,'  STA6SEM'
          outflow_from_sta6_to_wca=outflow_from_sta6-outflow
          outflow_from_sta6_to_semcyp=outflow
          sem_cyp_dmnd(jday)=sem_cyp_dmnd(jday)-outflow*1.9835
          jstr=jstr+1
          strflw(jstr,idss)=outflow
          gwdpth=(ells(nods140)-h(nods140))*s(nods140)
          IF (pond(nods140).ge.gwdpth) THEN
            stage=pond(nods140)-(ells(nods140)-h(nods140))*s(nods140)+
     &       ells(nods140)
          ELSE
            stage=(pond(nods140)-gwdpth)/s(nods140)+ells(nods140)
          END IF
          IF (stage.ge.10.8) THEN
            qcap=s140cap
          ELSE IF (stage.gt.10.5) THEN
            qcap=s140cap/0.3*(stage-10.5)
          ELSE
            qcap=0.0
          END IF
          IF (istaopt(10).eq.0) THEN
            addflowup=kflo(iflg89)
          ELSE
            addflowup=0.0
          END IF
c         IF (dryseas) THEN
c           frac_flow_c139=0.05
c         ELSE
c           frac_flow_c139=0.10
c         END IF
c         addflowup=addflowup+frac_flow_c139*(kflo(iflg89)+
c    &     amax1(kflo(ig155),0.0)+kflo(ig88))
          dble8p5 = dble(8.5)
          rmaxvl=(chdep(is140up)-dble8p5)*cl(is140up)*width(is140up)+
     &     cgseep(is140up,dble8p5,0,0.)+tovflc(is140up,dble8p5,0,0.,0,1)
     &     +addflowup*86400.+qu(is140up)
          rmaxvl=amax1(rmaxvl,0.0)/86400.
          qcap=amin1(qcap,rmaxvl)
          avail_cap_s140=s140cap-qcap
c         print *,avail_cap_s140,s140cap,qcap,stage,rmaxvl
c    &       ,'   AVAILCAPS140  ',chdep(is140up)
          IF ((.not.lwslo(inwcwca3a).or.(.not.lwslo(inwcwca3a)
```

```
     &    .and..not.lwslo_eaares(inwcwca3a).and.
     &    ((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no.gt.0)
     &    .or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.ieaarss_no.gt.0))
     &    .and.sta34_outf_flex_eaar_opt.eq.'YES'))
     &    .and.(lwslo(iwca3a_3a2).or.(lwslo_eaares(iwca3a_3a2)
     &    .and.((stg_eaa_res_2a.ge.stg_eaar_min_2a.and.ieaarsn_no.gt.0)
     &    .or.(stg_eaa_res_2b.ge.stg_eaar_min_2b.and.ieaarss_no.gt.0))
     &    .and.sta34_outf_flex_eaar_opt.eq.'YES')).and.wcaenv(iwca)
     &    .eq.'TRUE ') THEN
            outflow_from_sta6_to_s140=amin1(outflow_from_sta6_to_wca,
     &       avail_cap_s140)
            qu(is140up)=qu(is140up)+outflow_from_sta6_to_s140*86400.
            outflow_from_sta6_to_wca=outflow_from_sta6_to_wca-
     &       outflow_from_sta6_to_s140
            avail_cap_s140=avail_cap_s140-outflow_from_sta6_to_s140
c           print *,qu(is140up)/86400.,outflow_from_sta6_to_s140
c    &       ,avail_cap_s140,'  S140ST6'
          ELSE
            outflow_from_sta6_to_s140=0.0
          END IF
          DO j=1,ndspts(res_no,is)
            IF (inodopt(res_no,is,3).eq.1) THEN
              pond(nodepw(res_no,is,j))=pond(nodepw(res_no,is,j))+
     &         outflow_from_sta6_to_wca*86400./(ndspts(res_no,is)*gdar)
            ELSE
              qu(ircnl(res_no,is,j))=qu(ircnl(res_no,is,j))+
     &         outflow_from_sta6_to_wca*86400./ndspts(res_no,is)
            END IF
          END DO
        END IF
      END IF
      IF (resname(res_no) .NE. 'STA6 ' .AND. sfactor(res_no) .LE.
     &   sfactmin .AND. ires_small_sim(res_no) .EQ. 'YES') THEN
         res_struc_outflw(res_no) = res_struc_outflw(res_no)
     &    + outflow*1.9835
      ENDIF
      RETURN
      END
```

wsneeds.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rsantee $
C   $Id: wsneeds.F,v 1.14 2004/03/18 20:08:35 rsantee Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/wsneeds.F,v $
C   $Name: rel-5-5 $
C
C
C  THIS SUBROUTINE DETERMINES THE TOTAL SURFACE WATER REQUIREMENTS FOR WATER
SUPPLY
C  IN THE LOWER EAST COAST. THE NEED IS DEFINED AS THE VOLUME OF WATER REQUIRED
TO
C  MAINTAIN THE CANALS AT DESIRED MINIMUM LEVELS DURING DRY PERIODS. THE WATER,
IF
C  AVAILABLE, IS DELIVERED FROM THE APPROPRIATE WCA OR LAKE OKEECHOBEE.
C
C  THE NEEDS ARE DETERMINED FOR EVERY CANAL REACH TO BE MAINTAINED BY THE
STORAGE
C  AREAS.
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE wsneeds (nbranch, crmin0,jday, prevqu, rain, volwtr,
     & agrunoff_to_l8w, nsvarea, tempf_volwtr, rloc_seep_for_ws,
     & tempf_weircff, rloc_excess_for_ws, rloc_excss_to_canl_next_step,
     & rlv_seep_to_canl_c11b, crmin_bel_flr, regstg_wca, idn_wca_indx,
     & n_intcnl, ifloor_line_indx, iconvey_canal_no_wca, trig_wca_stg,
     & crmin, month_beg_min_mod_flr, month_end_min_mod_flr,
     & iday_beg_min_mod_flr, iday_end_min_mod_flr, numd,
     & ivary_slope_daily_opt,add_dmndagl8)

      INCLUDE 'wmm_par.inc'
      INCLUDE 'abc.inc'
      INCLUDE 'agdata.inc'
      INCLUDE 'c1.inc'
```

```
      INCLUDE 'c2.inc'
      INCLUDE 'routc.inc'
      INCLUDE 'stat.inc'
      INCLUDE 'wrout.inc'
      INCLUDE 'wcaot.inc'
      INCLUDE 'lakwca.inc'
      INCLUDE 'environ.inc'
      INCLUDE 'stormd.inc'

      DIMENSION canl_rf_avg(max_ncnls), crmin(max_ncnls,max_n_seas),
     & crmin_bel_flr(max_ncnls,max_n_seas),
     & crmin0(max_ncnls,max_n_seas), crmin1(max_ncnls,max_n_seas),
     & crmin3(max_ncnls,max_n_seas), iconvey_canal_no_wca(max_n_wcas),
     & idn_wca_indx(max_n_eaa_condt),
     & ifloor_line_indx(max_n_stor_w_oper_sch),
     & ilgth(max_ncnlloc), ivary_slope_daily_opt(max_ncnls),
     & lcmin_mod_floor(max_n_eaa_condt), n_intcnl(max_n_wcas),
     & nbranch(max_ncnls), nodec(max_ncnlloc), numd(12),
     & prevqu(max_ncnls), rain(max_ncells),
     & regstg_wca(max_n_wcas,max_n_zones_wca),
     & rf_in_canal(max_ncnls,15), rfin(max_ncnls),
     & rloc_excess_for_ws(max_n_serv_area),
     & rloc_excss_to_canl_next_step(max_ncnls),
     & rloc_seep_for_ws(max_n_serv_area), tempf_volwtr(max_n_res),
     & tempf_weircff(max_n_res,max_nostr_res), trig_wca_stg(max_n_wcas),
     & volwtr(max_n_res),prev_canl_rf_avg(max_ncnls)

      CHARACTER rlv_seep_to_canl_c11b*5

      LOGICAL lcmin_mod_floor, ltintrvl_min_mod_flr1,
     & ltintrvl_min_mod_flr2

      REAL net_flow

      DOUBLE PRECISION crmin,crmin0,CRMIN_bel_flr,crmin1,crmin3
     & ,stg_min_ws,RMNSTGC,vr,vr_secb,vr_firstb,VR_MAINB

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

C  DEFINE INDEX FOR L-8 CANAL - NEEDED FOR TAKING INTO ACCOUNT
C   PUMPAGE INTO M-CANAL

      IF (opt_l8_prop) THEN
         ICNL = IL8S
      ELSE
         ICNL = IL8
      ENDIF
      nmdays_rainavg0 = nmdays_rainavg0 + 1
      IF (nmdays_rainavg0 .gt. 14) nmdays_rainavg0 = 14
      IF (day1 .OR. nmdays_rainavg0 .EQ. 14) THEN
         do n = 1,nch
           canl_rf_avg(n) = 0.0
         enddo
      ENDIF
C
C
c  set minimum levels in canals which can be
```

```
c  dependent on WCA stages
c
      do n = 1,nch
        crmin1(n,iseas) = crmin0(n,iseas)
      enddo
c
c time dependent floor elevation for WCAs
c
      do iwca = 1,nwcas
         if (ifloor_line_indx(iwca+1) .gt. 0) then
           do i = 1,n_intcnl(iwca)
             rmnstg(iwca,i) = regstg_wca(iwca,ifloor_line_indx(iwca+1))
           enddo
         endif
      enddo
c
c  set condition for change in minimum canal levels from normal
c  operations
c
      julbeg_min_mod_flr = julian(month_beg_min_mod_flr
     $ ,iday_beg_min_mod_flr,numd)
      julend_min_mod_flr = julian(month_end_min_mod_flr
     $ ,iday_end_min_mod_flr,numd)
      LTINTRVL_min_mod_flr1 = (jday .ge. julbeg_min_mod_flr
     $ .and. jday .le. julend_min_mod_flr) .and.
     $ julend_min_mod_flr .gt. julbeg_min_mod_flr
      LTINTRVL_min_mod_flr2 = (jday .ge. julbeg_min_mod_flr
     $ .or. jday .le. julend_min_mod_flr) .and.
     $ julend_min_mod_flr .lt. julbeg_min_mod_flr
      LCMIN_mod_floor(is8bsn-6) = ((chdep(intcnl(iwca3a,
     $ iconvey_canal_no_wca(iwca3a))) .LT.
     $ rmnstg(idn_wca_indx(is8bsn-6),iconvey_canal_no_wca(iwca3a))
     $ + 0.05) .OR. (floor_grid_cell_opt .eq. 'MARSH' .AND.
     $ LWSLO_floor(iwca3a))) .AND. (LTINTRVL_min_mod_flr1
     $ .OR. LTINTRVL_min_mod_flr2)
      LCMIN_mod_floor(is7bsn-6) = ((trig_wca_stg(idn_wca_indx(is7bsn-6))
     $ .LT. rmnstg(idn_wca_indx(is7bsn-6),iconvey_canal_no_wca
     $ (iwca2a)) + 0.05 ) .OR. (floor_grid_cell_opt .eq. 'MARSH'
     $ .AND. LWSLO_floor(iwca2a))) .AND. (LTINTRVL_min_mod_flr1
     $ .OR. LTINTRVL_min_mod_flr2)
      LCMIN_mod_floor(is6bsn-6) = ((trig_wca_stg(idn_wca_indx(is6bsn-6))
     $ .LT. rmnstg(idn_wca_indx(is6bsn-6),iconvey_canal_no_wca
     $ (iwca1)) + 0.05) .OR. (floor_grid_cell_opt .eq. 'MARSH'
     $ .AND. LWSLO_floor(iwca1))) .AND. (LTINTRVL_min_mod_flr1
     $ .OR. LTINTRVL_min_mod_flr2)
c
c  WCA3A bel floor
      IF (LCMIN_mod_floor(is8bsn-6)) THEN
         DO n = 1,nch
           IF (iserv_area_loc(n) .eq. 3) THEN
              crmin1(n,iseas) = crmin_bel_flr(n,iseas)
           ENDIF
         ENDDO
      ENDIF
c
c  WCA2A bel floor
      IF (LCMIN_mod_floor(is7bsn-6)) THEN
```

```
         DO n = 1,nch
            IF (iserv_area_loc(n) .eq. 2) THEN
               crmin1(n,iseas) = crmin_bel_flr(n,iseas)
            ENDIF
         ENDDO
      ENDIF
c
c  WCA1 bel floor
      IF (LCMIN_mod_floor(is6bsn-6)) THEN
         DO n = 1,nch
            IF (iserv_area_loc(n) .eq. 1) THEN
               crmin1(n,iseas) = crmin_bel_flr(n,iseas)
            ENDIF
         ENDDO
      ENDIF
c
      do n = 1,nch
        crmin(n,iseas) = crmin1(n,iseas)
c       if (cnm(n) .eq. 'C111 ') print *,crmin(n,iseas)
c    $  ,LCMIN_mod_floor(is8bsn-6),iserv_area_loc(n)
c    $  ,'  BFLOOR  ',chdep(intcnl(iwca3a,
c    $  iconvey_canal_no_wca(iwca3a))),regstg_wca(idn_wca_indx(is8bsn-6)
c    $  ,ifloor_line_indx(idn_wca_indx(is8bsn-6)+1)) + 0.05
      enddo
c
c  check for periods of heavy rainfall during wet season
c
      npts=0
      do n = 1,nch
       rfin(n)=0.0
       DO 25 ic=1,nodcr(n)
         npts=npts+1
         nodec(ic)=nodecn(npts)/10
         ilgth(ic)=mod(nodecn(npts),10)
         rf=rain(nodec(ic))/12.
         rfin(n)=rfin(n)+rf*distnc(ilgth(ic))*width(n)
  25   CONTINUE
       IF (ivary_slope_daily_opt(n).eq.1) then
          IF (ndays_sim .GE. 14) THEN
             DO idr = 1,nmdays_rainavg0-1
                IF (ndays_sim .GT. 14) THEN
                   rf_in_canal(n,idr) = rf_in_canal(n,idr+1)
                ENDIF
                canl_rf_avg(n) = canl_rf_avg(n) + rf_in_canal(n,idr)
     &          /nmdays_rainavg0
             ENDDO
             rf_in_canal(n,nmdays_rainavg0)=rfin(n)/(width(n)*cl(n))*12.
             canl_rf_avg(n) =  canl_rf_avg(n) + rf_in_canal(n
     &       ,nmdays_rainavg0)/nmdays_rainavg0
          ELSE
             rf_in_canal(n,nmdays_rainavg0)=rfin(n)/(width(n)*cl(n))*12.
             canl_rf_avg(n) = rf_in_canal(n,nmdays_rainavg0)
          ENDIF
          icanal_drawdwn = 0
          do id = 1,idrwn
            IF (n.eq. icanal_no_drawdown(id)) icanal_drawdwn = 1
          enddo
```

```
          IF (ndays_prestorm_drawdown.eq.0 .OR. nstorm_day(ndays_sim_sd)
     &    .eq.0 .OR. icanal_drawdwn.eq.0 ) THEN
              rain_thres = 3.80
              IF (canl_rf_avg(n) .ge. rain_thres/14. .or.
     &         prev_canl_rf_avg(n) .ge. rain_thres/14.) then
                if (ws_status_cnl(n).eq.'FLWTH') then
                   crmin3(n,iseas) = -9.5
                else
                   crmin3(n,iseas) = -7.5
                endif
              else
                crmin3(n,iseas) = crmin(n,iseas)
              endif
              crmin(n,iseas) = crmin3(n,iseas)
          ENDIF
          prev_canl_rf_avg(n) = canl_rf_avg(n)
        ENDIF
c
        STG_MIN_WS_DSCHG(N) = CRMIN(N,iseas)
        IF (CNM(N) .EQ. canl_names_spec(2)) STG_MIN_WS_DSCHG(N) = MIN
     $  (CRMIN(N,iseas),12.0)
      ENDDO
      IF (NCNL_WS_MIN .GT. 0) THEN
         DO NC = 1,NCNL_WS_MIN
            STG_MIN_WS_DSCHG(NCL_NO(NC)) = (CRMIN(NCL_NO(NC),iseas) +
     $      CRMIN(NDS_CNL_NO(NC),iseas)) / 2.0
         END DO
      ENDIF
C  ISA DEFINES LOOP FOR SERVICE AREA
C  ISA = 1 : SERVICE AREA 1 (WCA-1)
C  ISA = 2 : SERVICE AREA 2 (WCA-2A)
C  ISA = 3 : SERVICE AREA 3 (WCA-3A)
C
C
      DO 100 ISA = 1,nsvarea
C
         rloc_seep_for_ws(isa) = 0.0
         rloc_excess_for_ws(isa) = 0.0
         IF (NSTART(ISA) .EQ. 0) GO TO 100
         ISTART = 1
         DSD = 0.0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C  ICMB : CANAL NUMBER FOR CANAL OF INTEREST ALONG MAIN BRANCH OF CANAL
C       NETWORK. INITIALLY IS THE MOST DOWNSTREAM REACH IN CANAL
C       NETWORK WHICH IS INPUT IN LECDEF FILE
C
   10    ICMB = ISTC(ISTART,ISA)
         DSDQU = DQU(ICMB)
C
C  LOOP FOR TOTAL NUMBER OF BRANCHES FOR CANAL NUMBER ICMB. THE NUMBER
C  OF BRANCHES INCLUDES CANAL ITSELF;THUS, THE NUMBER OF TRIBUTARIES
C  EQUALS NBRANCH(ICMB) - 1.
C
         total_ds_needs_trib = 0.0
   20    DO 90 IB = 1,NBRANCH(ICMB)
            IF (IB .GT. 1) THEN
```

```
C
C  THE FOLLOWING CODE IS EXECUTED ONLY IF CANAL NUMBER ICMB IN MAIN BRANCH
C  HAS TRIBUTARIES. THE NUMBER FOR THE MOST DOWNSTREAM CANAL MAINTAINED
C  IN EACH TRIBUTARY IS INPUT IN LECDEF IN ISTC ARRAY.
C
               ISTART = ISTART + 1
               ICTFST = ISTC(ISTART,ISA)
               WSNDFST = DQU(ICTFST)
               DSDB = 0.0
               total_ds_needs_trib1 = 0.0
   30          DO 70 IDB = 1,NBRANCH(ICTFST)
                  IF (IDB .GT. 1) THEN
C
C  THE FOLLOWING IS EXECUTED ONLY IF CANAL NUMBER ICTFST ALONG LEVEL ONE
C  TRIBUTARY HAS TRIBUTARIES
C
                     ISTART = ISTART + 1
                     ICTSCND = ISTC(ISTART,ISA)
                     WSNDSCND = DQU(ICTSCND)
                     DSDBD = 0.0
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
C  THE FOLLOWING IS EXECUTED ONLY IF CANAL NUMBER ICTSCND ALONG LEVEL TWO
C  TRIBUTARY HAS TRIBUTARIES
C
                     total_ds_needs_trib2 = 0.0
   40                DO 60 IDBB = 1,NBRANCH(ICTSCND)
                        IF (IDBB .GT. 1) THEN
                           ISTART = ISTART + 1
                           ICTTHRD = ISTC(ISTART,ISA)
                           WSNDTHRD = DQU(ICTTHRD)
                           DSDBDT = 0.0
   50                      IF (ws_status_cnl(ICTTHRD) .EQ. 'FLWTH') THEN
                              IF (nint(CRMIN(ICTTHRD,iseas)).EQ.-9 .OR.
     $                        nint(CRMIN(ICTTHRD,iseas)).EQ.-10) THEN
                                 VR = 0.0
                              ELSE
                                 VR=(CRMIN(ICTTHRD,iseas)-CHDEP(ICTTHRD)
     $                           )*CL(ICTTHRD) *WIDTH(ICTTHRD)-CGSEEP
     $                           (ICTTHRD,CRMIN(ICTTHRD,iseas) ,0,0.)
     $                           -TOVFLC(ICTTHRD,CRMIN(ICTTHRD,iseas),0,
     $                           0.,0,1)
                                 VR = VR - QU(ICTTHRD)
                                 VR = MIN(VR,0.0)
                              ENDIF
                           ELSE
                              VR = (CRMIN(ICTTHRD,iseas)-CHDEP(ICTTHRD))
     $                        *CL(ICTTHRD) *WIDTH(ICTTHRD) - CGSEEP
     $                        (ICTTHRD,CRMIN(ICTTHRD,iseas) ,0,0.)
     $                        -TOVFLC(ICTTHRD,CRMIN(ICTTHRD,iseas),0,0.,
     $                        0,1)
                              VR = VR - QU(ICTTHRD)
                           ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
                           IF (WSNDTHRD .LE. 0.0) VR = MAX(VR,0.0)
                           DSDBDT = DSDBDT + VR
```

```
                              DQU(ICTTHRD) = DSDBDT / 86400.
                              DQU(ICTTHRD) = AMAX1(DQU(ICTTHRD),0.0)
                              WSNDTHRD = DQU(ICTTHRD)
                              ICNLBT = ICTTHRD
                              ICTTHRD = iup_canal_ws(ICTTHRD)
                              IF (ICTTHRD .NE. ICTSCND) THEN
                                 GO TO 50
                              ELSE
                                 total_ds_needs_trib2 =
     $                           total_ds_needs_trib2 + amax1(DQU(ICNLBT),
     $                           0.0)
                                 DQU(ICTSCND) = DQU(ICTSCND) + AMAX1(DQU
     $                           (ICNLBT),0.0)
                                 DSDBD = DSDBD + AMAX1(DQU(ICNLBT),0.0) *
     $                           86400.
C
                                 IF (IDBB .EQ. NBRANCH(ICTSCND)) THEN
                                    DQU(ICTSCND) = AMAX1(DQU(ICTSCND),0.0)
                                    WSNDSCND = DQU(ICTSCND)
                                    IF (ifix(QU(ictscnd)) .GT. 0) THEN
                                       IF (ws_status_cnl(ictscnd) .EQ.
     $                                 'FLWTH') THEN
                                       seep_for_ws = min(QU(ICTSCND),abs
     $                                 (vr_secb) ,total_ds_needs_trib2
     $                                 * 86400.)
                                       rloc_for_ws = min
     $                                 (rloc_excss_to_canl_next_step
     $                                 (ictscnd),abs(vr_secb),
     $                                 total_ds_needs_trib2 * 86400.)
                                       rloc_seep_for_ws(isa) =
     $                                 rloc_seep_for_ws(isa) + amax1
     $                                 (seep_for_ws - rloc_for_ws,0.0)
                                       rloc_excess_for_ws(isa)=
     $                                 rloc_excess_for_ws(isa) +
     $                                 rloc_for_ws
                                       ELSE
                                       rloc_seep = amax1(QU(ictscnd) -
     $                                 rloc_excss_to_canl_next_step
     $                                 (ictscnd),0.0)
                                       total_need_for_seep = max
     $                                 (total_ds_needs_trib2 * 86400.
     $                                 +vr_secb,0.0)
                                       rem_need_for_seep = max
     $                                 (total_need_for_seep -
     $                                 rloc_excss_to_canl_next_step
     $                                 (ictscnd),0.0)
                                       rloc_excess_for_ws(isa)=
     $                                 rloc_excess_for_ws(isa) + amin1
     $                                 (total_need_for_seep ,
     $                                 rloc_excss_to_canl_next_step
     $                                 (ictscnd))
                                       rloc_seep_for_ws(isa) =
     $                                 rloc_seep_for_ws(isa) + amin1
     $                                 (rloc_seep,rem_need_for_seep)
                                       ENDIF
                                    ENDIF
                                 ENDIF
```

```
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                         ENDIF
                       ELSE
C
C  CALCULATE VOLUME OF WATER REQUIRED TO MAINTAIN CANAL ICTSCND
C  ALONG LEVEL TWO TRIBUTARY
C
                         IF (ws_status_cnl(ICTSCND) .EQ. 'FLWTH') THEN
                            IF (nint(CRMIN(ICTSCND,iseas)).EQ.-9 .OR.
     $                      nint(CRMIN(ICTSCND,iseas)).EQ.-10) THEN
                               VR = 0.0
                               vr_secb = 0.0
                            ELSE
                               VR = (CRMIN(ICTSCND,iseas)-CHDEP
     $                         (ICTSCND))*CL(ICTSCND) *WIDTH(ICTSCND)
     $                         - CGSEEP(ICTSCND,CRMIN(ICTSCND,iseas) ,
     $                         0,0.)-TOVFLC(ICTSCND,CRMIN(ICTSCND,
     $                         iseas),0,0.,0,1)
                               VR = VR - QU(ICTSCND)
                               VR = MIN(VR,0.0)
                               vr_secb = abs(vr)
                               IF (ifix(QU(ICTSCND)) .GT. 0 .AND.
     $                         nbranch(ictscnd) .EQ. 1) THEN
                                  total_need_for_seep = amax1(WSNDSCND
     $                            * 86400.,0.0)
                                  seep_for_ws = min(QU(ICTSCND),abs
     $                            (vr) ,total_need_for_seep)
                                  rloc_for_ws = min
     $                            (rloc_excss_to_canl_next_step
     $                            (ictscnd),abs(vr),
     $                            total_need_for_seep)
                                  rloc_seep_for_ws(isa) =
     $                            rloc_seep_for_ws(isa) + amax1
     $                            (seep_for_ws - rloc_for_ws,0.0)
                                  rloc_excess_for_ws(isa)=
     $                            rloc_excess_for_ws(isa) +
     $                            rloc_for_ws
                               ENDIF
                            ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                         ELSE
                            VR = (CRMIN(ICTSCND,iseas)-CHDEP(ICTSCND))
     $                      *CL(ICTSCND) *WIDTH(ICTSCND) - CGSEEP
     $                      (ICTSCND,CRMIN(ICTSCND,iseas) ,0,0.)
     $                      -TOVFLC(ICTSCND,CRMIN(ICTSCND,iseas),0,0.,
     $                      0,1)
                            vr_secb = vr
                            IF (ifix(QU(ICTSCND)) .GT.  0.AND. nbranch
     $                      (ictscnd) .EQ. 1) THEN
                               total_need_for_seep = max(WSNDSCND
     $                         * 86400. + VR,0.0)
                               rloc_seep = amax1(QU(ictscnd) -
     $                         rloc_excss_to_canl_next_step(ictscnd),
     $                         0.0)
                               rem_need_for_seep = amax1
     $                         (total_need_for_seep -
     $                         rloc_excss_to_canl_next_step(ictscnd),
```

```
     $                          0.0)
                                rloc_excess_for_ws(isa)=
     $                          rloc_excess_for_ws(isa) + amin1
     $                          (total_need_for_seep ,
     $                          rloc_excss_to_canl_next_step(ictscnd))
                                rloc_seep_for_ws(isa) =
     $                          rloc_seep_for_ws(isa) + amin1
     $                          (rloc_seep,rem_need_for_seep)
                             ENDIF
                             VR = VR - QU(ICTSCND)
                          ENDIF
C
                          IF (WSNDSCND .LE. 0.0) VR = MAX(VR,0.0)
                          DSDBD = DSDBD + VR
                          DQU(ICTSCND) = DSDBD / 86400.
                          IF (NBRANCH(ICTSCND) .EQ. 1)DQU(ICTSCND) =
     $                    AMAX1(DQU(ICTSCND),0.0)
                          WSNDSCND = DQU(ICTSCND)
                       ENDIF
   60                CONTINUE
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                     ICNLB = ICTSCND
                     ICTSCND = iup_canal_ws(ICTSCND)
                     IF (ICTSCND .NE. ICTFST) THEN
                        GO TO 40
                     ELSE
                        total_ds_needs_trib1 = total_ds_needs_trib1 +
     $                  amax1(DQU(ICNLB),0.0)
                        DQU(ICTFST) = DQU(ICTFST) + AMAX1(DQU(ICNLB),
     $                  0.0)
                        DSDB = DSDB + AMAX1(DQU(ICNLB),0.0) * 86400.
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                    IF (IDB .EQ. NBRANCH(ICTFST)) THEN
                        DQU(ICTFST) = AMAX1(DQU(ICTFST),0.0)
                        WSNDFST = DQU(ICTFST)
                        IF (ifix(QU(ictfst)) .GT. 0) THEN
                           IF (ws_status_cnl(ictfst) .EQ. 'FLWTH') THEN
                              seep_for_ws = min(QU(ICTFST),abs
     $                        (vr_firstb),total_ds_needs_trib1 * 86400.)
                              rloc_for_ws = min
     $                        (rloc_excss_to_canl_next_step (ictfst),abs
     $                        (vr_firstb),total_ds_needs_trib1 * 86400.)
                              rloc_seep_for_ws(isa) = rloc_seep_for_ws
     $                        (isa) + amax1(seep_for_ws - rloc_for_ws,
     $                        0.0)
                              rloc_excess_for_ws(isa)=
     $                        rloc_excess_for_ws(isa) + rloc_for_ws
                           ELSE
                              rloc_seep = amax1(QU(ictfst) -
     $                        rloc_excss_to_canl_next_step(ictfst),0.0)
                              total_need_for_seep = max
     $                        (total_ds_needs_trib1 * 86400.+vr_firstb,
     $                        0.0)
                              rem_need_for_seep = max
     $                        (total_need_for_seep -
     $                        rloc_excss_to_canl_next_step(ictfst),0.0)
                              rloc_excess_for_ws(isa)=
```

```
     $                        rloc_excess_for_ws(isa) + min
     $                        (total_need_for_seep ,
     $                        rloc_excss_to_canl_next_step(ictfst))
                              rloc_seep_for_ws(isa) = rloc_seep_for_ws
     $                        (isa) + amin1(rloc_seep,rem_need_for_seep)
                           ENDIF
                        ENDIF
                     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  ENDIF
               ELSE
C
C  CALCULATE VOLUME OF WATER REQUIRED TO MAINTAIN CANAL ICTFST IN LEVEL
C  ONE TRIBUTARY
C
                  IF (cnm(ictfst) .EQ. canl_names_spec(34) .AND.
     $            opt_flow_to_ts .EQ. 'TSRFPLN') THEN
                     IF (wca_import_variation .EQ. 'DAILY') THEN
                        stage_import_wca(il31w_min_rf) =
     $                  daily_stg_import_wca (ngage_index_env
     $                  (il31w_min_rf,1,1))
                     ELSE
                        IF (wca_import_variation .EQ. 'CONST') THEN
                           stage_import_wca(il31w_min_rf) =
     $                     avg_import_stage (il31w_min_rf)
                        ENDIF
                     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                     IF (main_pres_level_fl_prot_ts .EQ. 'TRUE ') THEN
                        stg_min_ws = min(stage_import_wca
     $                  (il31w_min_rf) -offset(il31w_min_rf,1),rmnstgs
     $                  (ictfst,1,iseas))
                     ELSE
                        stg_min_ws = stage_import_wca(il31w_min_rf) -
     $                  offset(il31w_min_rf,1)
                     ENDIF
                     stg_min_ws = max(stg_min_ws,crmin(ictfst,iseas))
                  ELSE
                     stg_min_ws = crmin(ictfst,iseas)
                  ENDIF
                  IF (ws_status_cnl(ICTFST) .EQ. 'FLWTH') THEN
                     IF (nint(crmin(ictfst,iseas)) .EQ. -9 .OR.
     $               nint(crmin(ictfst,iseas)) .EQ. -10) THEN
                        VR = 0.0
                        vr_firstb = vr
                     ELSE
                        VR = (stg_min_ws - CHDEP(ICTFST)) * CL(ICTFST) *
     $                  WIDTH (ICTFST) - CGSEEP(ICTFST,stg_min_ws,0,0.)
     $                  -TOVFLC (ICTFST,stg_min_ws,0,0.,0,1)
                        VR = VR - QU(ICTFST)
                        VR = MIN(VR,0.0)
                        vr_firstb = abs(vr)
                        IF (ifix(QU(ICTFST)) .GT. 0 .AND. nbranch
     $                  (ictfst) .EQ. 1) THEN
                           total_need_for_seep = amax1(WSNDFST * 86400.,
     $                     0.0)
                           seep_for_ws = min(QU(ICTFST),abs(vr) ,
```

```
     $                     total_need_for_seep)
                           rloc_for_ws = min
     $                     (rloc_excss_to_canl_next_step (ictfst),abs
     $                     (vr),total_need_for_seep)
                           rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa)
     $                     + amax1(seep_for_ws - rloc_for_ws,0.0)
                           rloc_excess_for_ws(isa)= rloc_excess_for_ws
     $                     (isa) + rloc_for_ws
                        ENDIF
                     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  ELSE
                     VR = (stg_min_ws - CHDEP(ICTFST)) * CL(ICTFST) *
     $               WIDTH (ICTFST) - CGSEEP(ICTFST,stg_min_ws,0,0.)
     $               -TOVFLC (ICTFST,stg_min_ws,0,0.,0,1)
                     vr_firstb = vr
                     IF (ifix(QU(ICTFST)) .GT.  0.AND. nbranch(ictfst)
     $               .EQ. 1) THEN
                        total_need_for_seep = max(WSNDFST * 86400.
     $                  + VR, 0.0)
                        rloc_seep = amax1(QU(ictfst) -
     $                  rloc_excss_to_canl_next_step(ictfst),0.0)
                        rem_need_for_seep = amax1(total_need_for_seep -
     $                  rloc_excss_to_canl_next_step(ictfst),0.0)
                        rloc_excess_for_ws(isa)= rloc_excess_for_ws(isa)
     $                  + amin1(total_need_for_seep ,
     $                  rloc_excss_to_canl_next_step(ictfst))
                        rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa) +
     $                  amin1(rloc_seep,rem_need_for_seep)
                     ENDIF
                     VR = VR - QU(ICTFST)
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  IF (opt_flow_to_ts .EQ. 'TSMINDL' .AND. cnm(ictfst)
     $            .EQ. canl_names_spec(34)) VR = VR + VOL332(MONTH)
     $            *86400.
c
                  IF (opt_flow_to_ts .NE. 'TSRFPLN' .AND. cnm(ictfst)
     $            .EQ. canl_names_spec(34) .AND. opt_flow_to_ts .NE.
     $            'TSMINDL') VR = VR + VOL332DO(MONTH) * 86400.
                  IF (opt_flow_to_ts .EQ. 'TSRFPLN' .AND. cnm(ictfst)
     $            .EQ.canl_names_spec(34)) THEN
                     IF (ngrid_cells_daily(itylsl) .GT. 0) THEN
                        depthgw = (ells(node_import_daily(itylsl,1)) - h
     $                  (node_import_daily(itylsl,1)))* s
     $                  (node_import_daily (itylsl,1))
                        IF (pond(node_import_daily(itylsl,1)) .GT.
     $                  depthgw) THEN
                           tylsl_stage = pond(node_import_daily
     $                     (itylsl,1)) - depthgw + ells
     $                     (node_import_daily(itylsl,1))
                        ELSE
                           tylsl_stage = (pond(node_import_daily
     $                     (itylsl,1)) - depthgw)/s(node_import_daily
     $                     (itylsl,1)) + ells(node_import_daily
     $                     (itylsl,1))
                        ENDIF
```

```
                     ELSE
                        tylsl_stage = 999.
                     ENDIF
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c
                  IF (opt_flow_to_ts .EQ. 'TSRFPLN' .AND. cnm(ictfst)
     $            .EQ.canl_names_spec(34)) THEN
                     IF (ngrid_cells_daily(itylsl) .GT. 0) THEN
                        IF (tylsl_stage .LT. daily_stg_import_wca
     $                  (ngage_index_env(itylsl,1,1))) THEN
                           excess = vr / 86400.
                           DSDB_excess = DSDB + excess * 86400.
                           DSDB_excess = DSDB_excess / 86400.
                        ENDIF
                     ENDIF
                  ENDIF
c
                  IF(WSNDFST .LE. 0.0 .AND. NBRANCH(ICTFST) .EQ. 1)
     $            VR =MAX(VR,0.0)
                  DSDB = DSDB + VR
                  IF (opt_flow_to_ts .EQ. 'TSRFPLN' .AND. cnm(ictfst)
     $            .EQ. canl_names_spec(34)) THEN
                     IF (ngrid_cells_daily(itylsl) .GT. 0) THEN
                        IF (tylsl_stage .LT. daily_stg_import_wca
     $                  (ngage_index_env(itylsl,1,1))) THEN
                           str_mx = DISCOEF(l31w,1,1,1)
                           DSDB = AMAX1((str_mx+DSDB_excess) * 86400.,
     $                     DSDB)
                        ENDIF
                     ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  ENDIF
                  DQU(ICTFST) = DSDB / 86400.
                  IF(NBRANCH(ICTFST) .EQ. 1) DQU(ICTFST) = AMAX1(DQU
     $            (ICTFST) ,0.0)
                  WSNDFST = DQU(ICTFST)
               ENDIF
   70       CONTINUE
            NCNL = ICTFST
C
C  GO TO UPSTREAM CANAL REACH
C
            ICTFST = iup_canal_ws(ICTFST)
C
C  CHECK IF NUMBER FOR UPSTREAM CANAL REACH IS EQUAL TO THAT FOR CANAL
C  IN MAIN MRANCH ( = ICMB). IF EQUAL,THE TOTAL WATER SUPPLY REQUIREMENTS
C  FOR LEVEL ONE TRIBUTARY HAS BEEN DETERMINED ,THUS CONTINUE ALONG MAIN
C  BRANCH. IF NOT EQUAL ,THE TOTAL REQUIREMENTS FOR LEVEL ONE TRIBUTARY
C  HAVE YET TO BE DETERMINED.
C
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
            IF (ICTFST .NE. ICMB) THEN
               DSDB = AMAX1(DSDB,0.0)
               GO TO 30
            ELSE
               DQU(ICMB) = DQU(ICMB) + AMAX1(DQU(NCNL),0.0)
```

```
                total_ds_needs_trib = total_ds_needs_trib + AMAX1(DQU
     $          (NCNL),0.0)
                DSD = DSD + AMAX1(DQU(NCNL),0.0) * 86400.
                IF(IB .EQ. NBRANCH(ICMB)) THEN
                   DQU(ICMB) = AMAX1(DQU(ICMB),0.0)
                   DSDQU = DQU(ICMB)
                   IF (ws_status_cnl(ICMB) .EQ. 'FLWTH') THEN
                      IF (ifix(QU(ICMB)) .GT. 0) THEN
                         seep_for_ws = min(QU(ICMB),abs(vr_mainb) ,
     $                   total_ds_needs_trib * 86400.)
                         rloc_for_ws = min
     $                   (rloc_excss_to_canl_next_step (icmb),abs
     $                   (vr_mainb),total_ds_needs_trib * 86400.)
                         rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa) +
     $                   amax1(seep_for_ws - rloc_for_ws,0.0)
                         rloc_excess_for_ws(isa)= rloc_excess_for_ws(isa)
     $                   + rloc_for_ws
                      ENDIF
                   ELSE
                      IF (ifix(QU(ICMB)) .GT. 0) THEN
                         rloc_seep = amax1(QU(icmb) -
     $                   rloc_excss_to_canl_next_step(icmb),0.0)
                         total_need_for_seep = max(total_ds_needs_trib
     $                   * 86400. + vr_mainb,0.0)
                         rem_need_for_seep = amax1(total_need_for_seep -
     $                   rloc_excss_to_canl_next_step(icmb),0.0)
                         rloc_excess_for_ws(isa)= rloc_excess_for_ws(isa)
     $                   + amin1(total_need_for_seep ,
     $                   rloc_excss_to_canl_next_step(icmb))
                         rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa) +
     $                   amin1(rloc_seep,rem_need_for_seep)
                      ENDIF
                   ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
c             IF (cnm(icmb) .EQ. 'C14  ')
c    $       print *,QU(ICMB)/86400.,VR_MAINB/86400.,total_ds_needs_trib
c    $            ,total_need_for_seep/86400.,'  BBBB'
                ENDIF
             ENDIF
          ELSE
C
C  CALCULATE VOLUME OF WATER REQUIRED TO MAINTAIN CANAL ICMB ALONG MAIN
C  BRANCH
C
C  FOR L-8 ADD PUMPAGE TO M-CANAL TO NEED
C
             IF (cnm(icmb) .EQ. canl_names_spec(24) .OR. cnm(icmb) .EQ.
     $       canl_names_spec(26) .OR. cnm(icmb) .EQ. canl_names_spec(25))
     $       THEN
                IF ( DMNDAGL8 .LT. 0.0) THEN
                   IF (opt_l8_prop) THEN
                      IF (ICMB .EQ. ICNL) THEN
                         NET_FLOW = qu(icmb)
                      ELSE
                         NET_FLOW = agrunoff_to_l8w
                      ENDIF
                   ELSE
```

```
                     NET_FLOW = qu(icmb)
                  ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                  WSAGL8 = 0.0
               ELSE
                  NET_FLOW = qu(icmb)
                  WSAGL8 = DMNDAGL8 / 1.9835
                  IF (opt_l8_prop) WSAGL8 = 0.0
               ENDIF
               IF (STAGELO .LT. CRMIN(ICMB,iseas)+HDC(ICMB,iseas) .AND.
     $         ((CNM(ICMB).EQ.canl_names_spec(24).AND. .NOT.opt_l8_prop)
     $         .OR. (CNM(ICMB).EQ.canl_names_spec(25).AND.opt_l8_prop)))
     $         THEN
                  RMNSTGC = MIN(CHDEP(ICMB),STAGELO-HDC(ICMB,iseas))
               ELSE
                  RMNSTGC = CRMIN(ICMB,iseas)
               ENDIF
C
               ADDNEED = WSAGL8 + add_dmndagl8/1.9835
               IF (NSTRCTR(ICMB) .GT. 0 .AND. runmode .EQ. 'SIMUL') THEN
                  TOT_PUMP = 0.0
                  ADDNEED = STROUT(RMNSTGC,ICM,ICMB,1,0,CRMIN ,VOLWTR,
     $            PREVQU,RAIN,0,TOT_PUMP,JDAY,tempf_volwtr ,
     $            tempf_weircff,0) + ADDNEED
               ELSE
                  ADDNEED = WSAGL8 + add_dmndagl8/1.9835
               ENDIF
               IF (runmode .EQ. 'CALIB') THEN
                  NET_FLOW = - (KFLO(IFLL8) + KFLO(IFLS2MC)) * 86400.
               ENDIF
            ELSEIF(cnm(ICMB) .EQ. canl_names_spec(27) .AND. runmode .EQ.
     $      'CALIB') THEN
               ADDNEED = 0.0
               RMNSTGC = CRMIN(ICMB,iseas)
               NET_FLOW = (KFLO(IFLS334)-KFLO(IFLS336)-KFLO(IFLS338)) *
     $         86400. + QU(ICMB)
            ELSEIF(cnm(ICMB) .EQ. canl_names_spec(28) .AND. runmode .EQ.
     $      'CALIB') THEN
               ADDNEED = 0.0
               RMNSTGC = CRMIN(ICMB,iseas)
               NET_FLOW = - KFLO(IFLS174) * 86400. + QU(ICMB)
            ELSEIF(cnm(ICMB) .EQ. canl_names_spec(42) .AND. runmode .EQ.
     $      'CALIB') THEN
               ADDNEED = 0.0
               RMNSTGC = CRMIN(ICMB,iseas)
               NET_FLOW = - KFLO(IFLS197) * 86400.
            ELSEIF (cnm(ICMB) .EQ. canl_names_spec(2)) THEN
               isstruc = 0
               DO is = 1,nstrctr(icmb)
                  isstruc = isstruc + 1
                  IF (flnm(istr(icmb,is)) .EQ. strname_spec(72))
     $            go to 80
               END DO
   80          ADDNEED = STRDCAP(IC18,isstruc,1,1)
               RMNSTGC = CRMIN(ICMB,iseas)
               NET_FLOW = QU(ICMB)
            ELSEIF (cnm(icmb) .EQ. 'C11  ') THEN
```

```
                TOT_PUMP = 0.0
                ADDNEED = 0.0
                RMNSTGC = CRMIN(ICMB,iseas)
                IF (rlv_seep_to_canl_c11b .NE. 'C11W ' .AND. DRYSEAS)
     $          THEN
                   DO is = 1,nstrctr(icmb)
                      ADDNEED = STROUT(RMNSTGC,ICMB,ICMB,1,0,CRMIN ,
     $                VOLWTR,PREVQU,RAIN,0,TOT_PUMP,JDAY,tempf_volwtr ,
     $                tempf_weircff,0) + ADDNEED
                   END DO
                ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
             ELSEIF (cnm(icmb) .EQ. 'US27S') THEN
                ADDNEED = 0.0
                RMNSTGC = CRMIN(ICMB,iseas)
                NET_FLOW = QU(ICMB) - outf_to_c9res_via_us27s * 86400.
             ELSE
                ADDNEED = 0.0
                RMNSTGC = CRMIN(ICMB,iseas)
                NET_FLOW = QU(ICMB)
             ENDIF
C
             IF (ws_status_cnl(ICMB) .EQ. 'FLWTH') THEN
                IF (nint(CRMIN(ICMB,iseas)) .EQ. -9 .OR.
     $          nint(CRMIN(ICMB,iseas)) .EQ. -10) THEN
                   VR = ADDNEED * 86400.
                   VR_MAINB = 0.0
                ELSE
                   VR = (RMNSTGC - CHDEP(ICMB))*CL(ICMB)*WIDTH(ICMB) -
     $             CGSEEP (ICMB,RMNSTGC,0,0.) - TOVFLC(ICMB,RMNSTGC,0,0.,
     $             0,1)
                   VR = VR - NET_FLOW
                   VR = MIN(VR,0.0) + ADDNEED * 86400.
                   VR_MAINB = abs(VR)
                   IF (ifix(QU(ICMB)) .GT. 0 .AND. nbranch(icmb) .EQ. 1)
     $             THEN
                      seep_for_ws = min(QU(ICMB),abs(vr) ,DSDQU*86400.)

                      rloc_for_ws = min(rloc_excss_to_canl_next_step
     $                (icmb),abs(vr),DSDQU * 86400.)
                      rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa) +
     $                amax1(seep_for_ws - rloc_for_ws,0.0)
                      rloc_excess_for_ws(isa)= rloc_excess_for_ws(isa) +
     $                rloc_for_ws
                   ENDIF
                ENDIF
             ELSE
                VR = (RMNSTGC - CHDEP(ICMB))*CL(ICMB)*WIDTH(ICMB) -
     $          CGSEEP (ICMB,RMNSTGC,0,0.) - TOVFLC(ICMB,RMNSTGC,0,0.,0,
     $          1) + ADDNEED * 86400.
                VR_MAINB = VR
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
                IF (ifix(QU(ICMB)) .GT. 0 .AND. nbranch(icmb) .EQ.1) THEN
                   total_need_for_seep = max(DSDQU*86400.+VR,0.0)
                   rloc_seep = max(QU(icmb) -
     $             rloc_excss_to_canl_next_step(icmb),0.0)
                   rem_need_for_seep = amax1(total_need_for_seep -
```

```
     $            rloc_excss_to_canl_next_step(icmb),0.0)
                  rloc_excess_for_ws(isa)= rloc_excess_for_ws(isa) +
     $            amin1(total_need_for_seep ,
     $            rloc_excss_to_canl_next_step(icmb))
                  rloc_seep_for_ws(isa) = rloc_seep_for_ws(isa) + amin1
     $            (rloc_seep,rem_need_for_seep)
               ENDIF
               VR = VR - NET_FLOW
            ENDIF
            IF (cnm(icmb) .EQ. canl_names_spec(42)) vr = max(vr ,
     $      vol18c(month))
            IF(DSDQU.LE. 0.0 .AND. NBRANCH(ICMB) .EQ. 1) VR = MAX
     $      (VR,0.0)
            DSD = DSD + VR
            DQU(ICMB) = DSD / 86400.
            IF(NBRANCH(ICMB) .EQ. 1) DQU(ICMB) = AMAX1(DQU(ICMB),0.0)
            DSDQU = DQU(ICMB)
         ENDIF
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
C
   90 CONTINUE
      IF (iup_canal_ws(ICMB) .GT. nch) THEN
         DSD = 0.0
         DQU(ICMB) = AMAX1(DQU(ICMB),0.0)
         IF(ISTART .EQ. NSTART(ISA)) GO TO 100
         ISTART = ISTART + 1
         GO TO 10
      ELSE
         DSD = AMAX1(DSD,0.0)
         ICMB = iup_canal_ws(ICMB)
         GO TO 20
      ENDIF
  100 CONTINUE
      DO INC = 1,NCH
      IF (DQU(INC) .LT. 0.0) DQU(INC) = 0.0
      DQU_ORIG(INC) = DQU(INC)
      END DO
c      write(445,'(i5,i4)') month,iday
c      write(445,900) (DQU(I),I=1,NCH)
c900   FORMAT(30F7.0)
      RETURN
      END
C---5----0----5----0----5----0----5----0----5----0----5----0----5----0-2
```

xy.F

```
C-------------------------------------------------------------------------------
C
C DISCLAIMER:
C
C ANY INFORMATION, INCLUDING BUT NOT LIMITED TO SOFTWARE AND DATA,
C RECEIVED FROM THE SOUTH FLORIDA WATER MANAGEMENT DISTRICT ("DISTRICT")
C IN FULFILLMENT OF A PUBLIC RECORDS REQUEST IS PROVIDED "AS IS" WITHOUT
C WARRANTY OF ANY KIND, AND THE DISTRICT EXPRESSLY DISCLAIMS ALL EXPRESS
C AND IMPLIED WARRANTIES, INCLUDING BUT NOT LIMITED TO THE IMPLIED
C WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.
C THE DISTRICT DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS
C REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE INFORMATION
C PROVIDED TO YOU BY THE DISTRICT IN TERMS OF CORRECTNESS, ACCURACY,
C RELIABILITY, TIMELINESS OR OTHERWISE.  THE ENTIRE RISK AS TO THE
C RESULTS AND PERFORMANCE OF ANY INFORMATION OBTAINED FROM THE DISTRICT
C IS ENTIRELY ASSUMED BY THE RECIPIENT.
C
C-------------------------------------------------------------------------------
C
C   CVS Keywords
C   $Author: rnovoa $
C   $Id: xy.F,v 1.4 2003/07/03 20:07:53 rnovoa Exp $
C   $Source: /vol/hsm2/cvsroot/models/WMM2K/src/xy.F,v $
C   $Name: rel-5-5 $
C
C-------------------------------------------------------------------------------

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      SUBROUTINE xy (minx, maxx, isum, node, ix, iy)

      INCLUDE 'wmm_par.inc'

      DIMENSION isum(max_n_rows), maxx(max_n_rows), minx(max_n_rows)

C---5----0----5----0----5----0----5----0----5----0----5----0----5----0----5----0-2

      IY = 1
   10 IF( NODE .GE. ISUM(IY) .AND. NODE .LE. ISUM(IY+1)) THEN
         IX = NODE - MINX(IY) - 1 - ISUM(IY)
         RETURN
      ELSE
         IY = IY + 1
         IF( IY .GT. 66) THEN
            PRINT *, 'ERROR IN UTILITY XY '
            PRINT *, 'JOB ABORTED '
            STOP 11
         ENDIF
         GOTO 10
      ENDIF
      END
```