Appendix M. Fortran Source Code for GEMSS Modules

WQCBM Module

```
! ******************    wq3dcb_Dino.f90      **********************
! WQ3DCB_Dino - 3-D Water Quality (Carbon-Based) Model for Budd Inlet
! Two algae groups included - Diatoms and Dinos 9/20/99
! Integrating into JEEAI's 3D model GLLVHT
! Coded by Dr. Jian Wu,
! QA by Dr. J. E. Edinger & Dr. Venkat Kolluru
! Designed by Mr. Chuck Boatman
! January 19-30, 1998
!
! J. E. Edinger Associates, Inc.
! 37 West Avenue
! Wayne, Pennsylvania 19087-3226
! phone: 610 293 0757
! fax: 610 293 0965
! web: www.jeeai.com
!
! Coding Notes for WQ3DCB_BI (1/19/98)
! 1) replace CBOD_D (dissolved) and CBOD_P (particulate) with
!    CBOD_F (fast reacting) and CBOD_S (slow reacting);
! 2) add three forms of particulate organic carbon
!    - fast reacting, slow reacting and refractory;
! 3) chnaged the excretion loss term to be based on (gp-dr);
! 4) phosphorus mineralization is derived only from dissolved OP;
! 5) add hydrolysis process to account conversion of particulate
!    form to dissolved form
! 6) using measured sediment fluxs for SOD, NH3, NO3 and PO4
!
! Coding Notes for WQ3D_BI (12/4/97)
! (mainly based on Chuck Boatman, Aura Nova; 11/30/97 e-mail)
! 1) divide CBOD, ON, OP into dissolved (CBOD_D, ON_D, OP_D) and
!    particulate (CBOD_P, ON_D, OP_D) components which make
!    verification easier and kinetics more accurate;
! 2) settling term is only applied for particulate components;
! 3) separate phytoplankton losses other than zooplankton grazing
!    into three components: respiration, death and excretion and cycle
!    them into proper state variables;
! 4) phytoplankton excretion is added in WQ3D_BI model. It is recycled
!    to the dissolved nutrients;
! 5) cycle only phytoplankton death to particulate nutrient pools;
! 6) cycle phytoplankton loss due to respiration to ammonia and
!    inorganic phosphorus;
! 7) include a fraction (fe) which is fraction of PP which is
!    excreted as DOM and is added to the DOC, DON and DOP pools;
! 8) combined the dissolved and particulate forms in mineralization
!    and oxidation terms;
! 9) the denitrification for CBOD is only applied for the dissolved
!    component CBOD_D;
!10) take out the (1-fD7) and (1-fD8) in the settling loss terms;
!11) the fractions (foc), (fon) and (fop) now represent the fraction
!    of dissolved organic matter released during death;
!12) add sources from micro-zooplankton grazing to particulate
!    nutrient pools;
!13) new SOD, SEDnh3 relationships based on new field data
!    from Budd Inlet Studies.
```

```
!
! Coding Notes for GLLVHT_EUTRO  (10/1/97)
! Items are further detailed in JEEAI 10/10/97 memo
! on Review and Formulation of Eutro5 for Use in the Three
! Dimensional GLLVHT Model and Budd Inlet Studies
! 1)  the code is based on the corrected and improved version of
!     the formulae in the WASP5 Manual per JEEAI memo drafted 10/10/97;
! 2)  the notation follows those in GLLVHT; and in most cases, the same
!     symbols as in Eutro5 equations are used for rate and coefficient ;
! 3)  the surface source/sink terms are first presented before the k
loop,
!     and sediment exchange source/sink terms are followed after the k
loop;
! 4)  settling terms are properly coded in the 3-D model frame (not in
WASP5);
! 5)  use Di Toro et al. (1971) & Boatman (1984)'s depth-dependent light
!     limiting function for stratified waterbody;
! 6)  use Boatman (1997) zooplankton grazing;
! 7)  separate zooplankton graze from the algae respiration/death term;
! 8)  only algae respiration/death (not zooplankton graze) is included in
!     the nitrogen and phosphorus cycles (JEEAI 10/10/97 memo) ;
! 9)  use Wanninkhof formula as developed in Boatman(1997)
!     for DO reaeation coefficient;
! 10) use Pamatmat(1971)'s work on Budd Inlet for sediment oxygen demand;
! 11) use Pamatmat(1971)'s work on Budd Inlet for sediment release of
ammonia;
!
! Definition of source/sink terms in GLLVHT is h(n,k,nc)
!
!     where I = index for x-direction  [m1 is 1-D array for 2D (i,j)]
!           J = index for y-direction
!           k = index for z-direction
!           nc= index for water quality constituents
!
!     Hydrodynamic/Transport Consitutent (GLLVHT)
!
!       I_Temp   = Temperature                                  (GLLVHT)
!       I_Saln   = Salinity                                     (GLLVHT)
!
!     Water Quality State Variables (WQ3D)
WQ3DCB_BI
!
(X- modeled)
!       I_NH3    =   Ammonia Nitrogen,                           g N/m^3
(C1)    X
!       I_N03    =   Nitrate Nitrogen,                           g N/m^3
(C2)    X
!       I_PO4    =   Inorganic Phosphorus,                       g P/m^3
(C3)    X
!       I_Phyt   =   Phytoplankton Carbon,                       g C/m^3
(C4)    X
!       I_CBOD   =   Carbonaceous BOD,                           g O2/m^3
(C5)
```

```
!       I_DO     =   Dissolved Oxygen,                              g O2/m^3
(C6)    X
!       I_ON     =   Organic Nitrogen,                              g N/m^3
(C7)
!       I_OP     =   Organic Phosphorus,                            g P/m^3
(C8)
!------ I_CBOD_D =   Dissolved Carbonaceous BOD,                    g O2/m^3
(C9)
! 6 new I_CBOD_P =   Particulate Carbonaceous BOD,                  g O2/m^3
(C10)
!       I_ON_D   =   Dissolved Organic Nitrogen,                    g N/m^3
(C11)   X
!       I_ON_P   =   Particulate Organic Nitrogen,                  g N/m^3
(C12)   X
!       I_OP_D   =   Dissolved Organic Phosphorus,                  g P/m^3
(C13)   X
!       I_OP_P   =   Particulate Organic Phosphorus,                g P/m^3
(C14)   X
!------ I_CBOD_F =   Fast Reating Dissolved Carbonaceous BOD,       g O2/m^3
(C15)   X
! 5 new I_CBOD_S =   Slow Reating Dissolved Carbonaceous BOD,       g O2/m^3
(C16)   X
!       I_OC_P_F =   Fast Reacting Particulate Organic Carbon,      g C/m^3
(C17)   X
!       I_OC_P_S =   Slow Reacting Particulate Organic Carbon,      g C/m^3
(C18)   X
!       I_OC_P_R =   Refractory Particulate Organic Carbon,         g C/m^3
(C19)   X
!------
! 2 new I_DAP    =   Diatoms algae group,                          g C/m^3
(C20)
!       I_DFP    =   Dinos algae group,                            g C/m^3
(C20)
!
Subroutine WQCBM3DSSTerms
      Use dfwin
      Use G7allocmemory
      Use AllocMemoryWQCBM3DVariables
      Use GEMSS_GAMAllocMemoryVariables
      Include 'G7variables.f90'
      Include 'G7WQCBMVariables.f90'
      Include 'GEMSS_GAMVariables.f90'
      !
      !Local variables
      Integer(2)  i, j, k, nc, AlgLoop, id
      Integer(4)  n
      Real(8)          depth,depth1,d1,temp20,tempk
      Real(8)          csat,k2,Thtk2
      Real(8)          sod,sedno3,sednh3,sedpo4,plimit, Sum
      !
      !Declare parameters and constants - WQ3DCB_BI
      Real(8)          anc,k71,th71,kmnc,k12,th12,knit,sednh3m, snh3
! rate/cons. for C1 - NH3
      Byte       isnh3type
```

```
      Real(8)          k2d,th2d,kno3,sedno3m
! rate/cons. for C2 - NO3
      Real(8)          apc,k83,th83,kmpc,sedpo4m,plc
! rate/cons. for C3 - PO4
      Real(8)          thkt, &       ! rate/cons. for C4 - PHYT
                       th1c,kmp,th1r,fe,as
      Real(8)          c2chla_d,rins_d,kmn_d,kgmicro_d,kgmacro_d,k1d_d, &
! rate/cons. for C4 - PHYT
                       k1c_d,k1r_d,vs4_d, thkt_d, th1c_d, kmp_d, th1r_d,
&
                       fe_d, as_d, th1d_d
      Byte       ZPGMode_d
      Real(8)          c2chla_f,rins_f,kmn_f,kgmicro_f,kgmacro_f,k1d_f, &
! rate/cons. for C4 - PHYT
                       k1c_f,k1r_f,vs4_f, thkt_f, th1c_f, kmp_f, th1r_f,
&
                       fe_f, as_f, th1d_f
      Byte       ZPGMode_f
      Real(8)          kdf,kds,sodm,thsod, sp4, sp5, sp7,
ReaerationFactor ! rate/cons. for C6 - DO
      Real(8)          aoc,thd,kbod,foc
! rate/cons. for C9  - CBOD_F
      Real(8)          fd5
! rate/cons. for C10 - CBOD_S
      Real(8)          kh7p,thh7p,fon
! rate/cons. for C11 - ON_D
      Real(8)          vs7,ancp
! rate/cons. for C12 - ON_P
      Real(8)          kh8p,thh8p,fop
! rate/cons. for C13 - OP_D
      Real(8)          vs8,apcp
! rate/cons. for C14 - OP_P
      Real(8)          fd9f,fg9f,kpd9f,thpd9f,vs9
! rate/cons. for C17 - OC_P_F
      Real(8)          fd9s,fg9s,kpd9s,thpd9s
! rate/cons. for C18 - OC_P_S
      Real(8)          fd9r,fg9r                                    !
rate/cons. for C19 - OC_P_R

      Real(8)          vgrav
      Real(8)          xri_d, xri_f, pnh3_f
      Real(8)          gmicro_d, gmacro_d, gmicro_f, gmacro_f
      Real(8)          ggraze_d, ggraze_f, gp_d, gp_f, dr_d, dr_f, dd_d,
dd_f
      Integer(2) option_fnh3
      Byte       DFPStatus, DAPStatus, SDOERMethod
      !
      !Variables for time varying rates and constants
      Real(8)          Value1, Value2, Factor, Value, PhytoConc,
RateConversion,VelocityConversion
      Integer(2) NumTVRCFiles, jd, jdd, nci
      Byte       VersionCheck, BytVal1, BytVal2, BytVal3, GAMRegionNum
      Integer(4) iretw, istat, DTTM2Julian, ierr
      !
```

```
      !Each term in the consituent equation
      Real(8)          p01a,p01b,p01c,p01d,p02a,p02b,p02c,p03a,p03b,p03c
      Real(8)          p06a,p06b,
p06c,p06d,p15a,p15b,p15c,p15d,p15e,p16a,p16c,p16d,p16e
      Real(8)
      p10a,p10b,p10c,p10d,p11a,p11b,p11c,p11d,p13a,p13b,p13c,p13d
      Real(8)
      p14a,p14b,p14c,p14d,p14e,p17a,p17b,p17c,p17d,p18a,p18b,p18c,p18d,p1
9a,p19b,p19c
      Real(8)          kinhib,fnh3,k2_wind
      Real(8)          epilson
      Real(8)
      p41a,p41b,p41c,p41d,p41e,p41f,p42a,p42b,p42c,p42d,p42e,p42f
      Logical(1)  ReadDefaultSettingsOnce
      !
      !Temperature and light dependent velocity variables
      Byte       UseVtemp_f
      Real(8)          Vtmax_f
      Real(8)          b_f
      Real(8)          c_f
      Real(8)          tL_f
      Real(8)          tH_f
      Real(8)          Voff_f
      Byte       UseVlight_f
      Real(8)          Vlmax_f
      Real(8)          Alpha_f

      !New Variable for reaeration equation
      Real(8)          DMO2, TempWad10

      Byte       InsideGAM

      Real(8)          aoc1, alpha0, alpha1
      Real(4)          Ke_a, Ke_b, Ke_c

      !SP 05/16/2008
      Real(8) foc_WQADD, fd5_WQADD, fd9f_WQADD, fd9s_WQADD, fd9r_WQADD,
fon_WQADD, fop_WQADD
      !End SP 05/16/2008
      !
      !This line should be there in all water quality models
      Common/CommonVariablesForAllWQModels/&
                 i, j, k, n, nc, d1, aoc1, &
                 foc_WQADD, fd5_WQADD, fd9f_WQADD, fd9s_WQADD,
fd9r_WQADD, fon_WQADD, fop_WQADD
      !
      !GAM and WQCBM
      Common/CommonVariablesBetweenGAMandWQCBM/&
                 apc, anc, ancp, apcp, aoc,&
                 fd9f, fg9f, fd9s, fg9s, fd9r, fg9r, InsideGAM

      Common /WQCBMLocalVariables/&
                 depth,depth1,temp20,tempk,&
                 csat,Thtk2,&
```

```
              sod,sedno3,sednh3,sedpo4,plimit,&
              k71,th71,kmnc,k12,th12,knit,sednh3m, snh3,&
              isnh3type,&
              k2d,th2d,kno3,sedno3m,&
              k83,th83,kmpc,sedpo4m,plc,&
              thkt, &
              th1c,kmp,th1r,fe,as,&
              c2chla_d,rins_d,kmn_d,kgmicro_d,kgmacro_d,k1d_d, &
              k1c_d,k1r_d,vs4_d, thkt_d, th1c_d, kmp_d, th1r_d, &
              fe_d, as_d,&
              ZPGMode_d,&
              c2chla_f,rins_f,kmn_f,kgmicro_f,kgmacro_f,k1d_f, &
              k1c_f,k1r_f,vs4_f, thkt_f, th1c_f, kmp_f, th1r_f, &
              fe_f, as_f, &
              ZPGMode_f,&
              kdf,kds,sodm,thsod, sp4, sp5, sp7, ReaerationFactor,&
              thd,kbod,foc,&
              fd5,&
              kh7p,thh7p,fon,&
              vs7,&
              kh8p,thh8p,fop,&
              vs8,&
              kpd9f,thpd9f,vs9,&
              kpd9s,thpd9s,&
              vgrav, &
              xri_d, xri_f, pnh3_f, &
              gmicro_d, gmacro_d, gmicro_f, gmacro_f,&
              ggraze_d, ggraze_f, gp_d, gp_f, dr_d, dr_f, dd_d, dd_f,&
              option_fnh3,&
              Value1, Value2, Factor, Value, &
              NumTVRCFiles, ierr, &
              VersionCheck, BytVal1, BytVal2, BytVal3,&
              iretw, istat,&
              UseVtemp_f,&
              Vtmax_f,&
              b_f,&
              c_f,&
              tL_f,&
              tH_f,&
              Voff_f,&
              UseVlight_f,&
              Vlmax_f,&
              Alpha_f,PhytoConc,DAPStatus,DFPStatus,SDOERMethod,&
              Ke_a, Ke_b, Ke_c, th1d_d, th1d_f

Data ReadDefaultSettingsOnce/.True./
Save

!
!Set local default waterquality rates and constants
jdd = 1
InsideGAM = 0
If(NumWQCBM3DRegions == 0) Then
      If(ReadDefaultSettingsOnce) Then
```

```
                  Call SetWQCBM3DRatesAndConstantsLocalDefault
                  ReadDefaultSettingsOnce = .False.
            End If
      End If
      Call ReadWQCBM3DTVRCFiles(jdd)
      !
      !Re-initilization at the beginning of the time step
      WQCBM3D_SumPhyt = 0.0
      WQCBM3D_vlight  = 0.0
      WQCBM3D_VTemp   = 0.0
      !vlmax   = 0.35e-4     ! (m/s)
      !vtmax   = 1.45e-4     ! (m/s)
      !voff    = 0.35e-4     ! (m/s)
      !
      !alphax  = 10.00       ! (um m2 uEinst)
      vgrav   = vs4_f        ! (m/s)
      If(mdltime >= 56671560)Then
            Continue
      End If
      !
      ! loop start for each control volume
      Do n = 1, n1
            If(nm(n) /= 1) Cycle
            i = i1(n)
            j = j1(n)
            !
            !Get spatial and time varying rates and constants
            If(NumWQCBM3DRegions /= 0) Then
                  If(WQCBM3DRegionStatus(i,j) /= 0 .and.
WQCBM3DRegionRCDStatus(WQCBM3DRegionStatus(i,j)) == 1) Then
                        Call SetWQCBM3DRatesAndConstantsLocalChange
                        jdd = WQCBM3DRegionStatus(i,j) + 1
                  Else
                        Call SetWQCBM3DRatesAndConstantsLocalDefault
                        jdd = 1
                  End If
            End If
            !
            !VSK: 06-24-2010 - To handle values coming from met spatially
varying boundary condition
            If(MaxNumMetBcs > 0) Then
                  PAR = MetBCPAR(dsgdr(CellMetBCNum(n)))
                  Albedo = MetBCAlbedo(dsgdr(CellMetBCNum(n)))
                  SolRad = MetBCrs(dsgdr(CellMetBCNum(n)))
            End If
            !
            !Update constant values with time dependent values
            Call ReadWQCBM3DTVRCFiles(jdd)

            If(n == 2376)Then
                  Continue
            End If
            depth = 0.0
            kt = ktwb(cwbn(i,j))
```

```
            d1 = dzk(kt) - z(n)
            CellArea = dxx(i,j)*dyy(i,j)
            !
            !DO reaeration through the surface layer (I_DO, C6)
            tempk = c(n,kt,I_Temp)
            !if(tempk < 0)tempk = 0.d0
            tempk = tempk + 273.15
            csat = dexp(-139.34411+(1.575701e+05/tempk)-(6.642308e+07 &
                  /(tempk**2))+(1.243800e+10/(tempk**3))-(8.621949e+11 &
                  /(tempk**4))-(c(n,kt,I_Saln)/1.80655)*(3.1929e-02 &
                  -(19.428/tempk)+(3.8673e+03/(tempk**2))))
! saturation DO (g m-3)

            !csat = csat + 3.0

            Call ComputeReAerationCoefficientInWQCBM

            !SP 09/11/2007 Reaeration Scaling when there is Ice present
            If(UseIGModel) k2T = k2T*SurfaceReAerScaleIce(n)
            !End SP 09/11/2007 Reaeration Scaling when there is Ice
present

            k2T = k2T*ReaerationFactor

            h(n,kt,I_DO) = acc(I_DO)*(h(n,kt,I_DO)+k2T*(csat-
c(n,kt,I_DO))*CellArea) !DO reaeration
            !
            ! ----- compute Dinos's vlight and WQCBM3D_VTemp for all
layers for given n
            ! reinitilization of ri(k) for the next n-loop and next time
step
            If(UseLOTTMethod == 1) Then
                  WQCBM3D_ri = 0.0
                  Do k = kt, k0(i,j)
                        !
                        !Total phytoplankton concentration
                        PhytoConc = c(n,k,I_DAP) + c(n,k,I_DFP)
                        If(GenAlgaeModel)Then
                              Do AlgLoop = 1, ngamcs
                                    id = I_GAM(AlgLoop)
                                    PhytoConc = PhytoConc + c(n,k,id)

                              End Do !AlgLoop
                        End If
                        depth = depth + d1
! depth from the surface
                        depth1 = depth - d1/2.0
                        WQCBM3D_SumPhyt(n) = WQCBM3D_SumPhyt(n) +
PhytoConc*d1                              ! total algae
                        ke       = 0.510 +
0.0023*WQCBM3D_SumPhyt(n)/c2chla_d                              !
extenction coeff.(m-1)
                        LEC(n,k) = ke
```

```
                        WQCBM3D_ri(k)  = solrad*0.2391d+00*dexp(-
1.*ke*depth1)                                ! radiation at the layer
                        If (WQCBM3D_ri(k) < 1.0e-10) WQCBM3D_ri(k) = 0.0
                        If(UseVlight_f == 1) WQCBM3D_vlight(n,k) =
Vlmax_f*(dtanh(Alpha_f*WQCBM3D_ri(k)*4.15/(Vlmax_f*1.0e6)))
                        !
                        ! a new formula from Chuck 10/7/99
                        If(UseVtemp_f == 1) WQCBM3D_VTemp(n,k)  =
Vtmax_f*(1.0 - dexp(-b_f*(c(n,k,I_Temp) - tL_f)))* &
                                                              (1.0 - dexp(-
c_f*(tH_f - c(n,k,I_Temp)))) - voff_f
                        Value = WQCBM3D_vlight(n,k)
                        WQCBM3D_vlight(n,k)= value * 1.5
                        Value = WQCBM3D_VTemp(n,k)
                        WQCBM3D_VTemp(n,k)= value * 1.5
                        If(WQCBM3D_VTemp(n,k) .lt. 0.0) WQCBM3D_VTemp(n,k)
= 0.0
                        d1 = dzk(k+1)
                  End Do !k
            !
            !GP and VSK: 04-06-2010
            Else
                  Do k = kt, k0(i,j)
                        !
                        !Calculate total chlorophyll a across all
phytoplankton groups in WQCBM and GAM
                        !New variable TotChla total chl a from all phyto
groups (ugA/L)
                        WQCBM3D_TotChla(n,k) = 0
                        If(GenAlgaeModel)Then
                              Do AlgLoop = 1, ngamcs
                                    id = I_GAM(AlgLoop)
                                    !
                                    !VSK: 05-05-2010
                                    GAMRegionNum = 1
                                    If(NUMGAM3DRegions > 0) GAMRegionNum =
GAM3DRegionStatus(i,j)  + 1
                                    !
                                    !factor of 1000/cchl in next eqn
converts mgC/L to ugA/L. Need code for correct region numbers
                                    WQCBM3D_TotChla(n,k) =
WQCBM3D_TotChla(n,k) + c(n,k,id)*1000.0/GAM3D_cchl(id-
ncGAMst+1,GAMRegionNum) !ugA/L
                              End Do !AlgLoop
                         End If
                        WQCBM3D_TotChla(n,k) = WQCBM3D_TotChla(n,k) +
c(n,k,I_DAP)*1000.0/c2chla_d     !ugA/L
                        WQCBM3D_TotChla(n,k) = WQCBM3D_TotChla(n,k) +
c(n,k,I_DFP)*1000.0/c2chla_f     !ugA/L
                        !
                        !Calculate light extinction variables
                        !LEC is light extinction coefficient (m^-1)
                        !KeCalc is the product of LEC and layer thickness
for use in later eqns in WQCBM and GAM (dim'less)
```

```
                        !RExtPar is the fraction of PAR that is attenuated
in this layer for use in later eqns in WQCBM and GAM (dim'less)
                        LEC(n,k) = Ke_a + Ke_b*WQCBM3D_TotChla(n,k)**Ke_c
!light extinction (m^-1)
                        !d1 is layer thickness (m). For the surface layer
d1=dzk(kt)-z(n). For all other layers d1=dzk(k)
                        KeCalc(n,k) = LEC(n,k)*d1
                        RExtPAR(n,k) = exp(-KeCalc(n,k))
                        !
                        !if GEMSS previously accounted for reflection in
the estimate of solrad then (1-Albedo) should not be in this eqn
                        If (k == kt) then
                              RTopPAR(n,k) = PAR*SolRad*(1.0 - Albedo)
!new variables RTopPAR, PAR, and Albedo used for PAR intensity at top of
this layer (W/m^2)
                        Else
                              RTopPAR(n,k) = RBotPAR(n,k-1)
                        EndIf
                        RBotPAR(n,k) = RTopPAR(n,k) * RExtPAR(n,k)    !new
variable RTopPar is PAR intensity at the bottom of this layer (W/m^2)
                        !depth-integrated average PAR light in layer (n,k)
                        !over layer depth per Chapra eqn 27.8
                        WQCBM3D_ri(k) = RTopPAR(n,k) / KeCalc(n,k) * (1.0
- RExtPAR(n,k))
                  !same as original lines 359-372
                        If (WQCBM3D_ri(k) < 1.0e-10) WQCBM3D_ri(k) = 0.0
                  If(UseVlight_f == 1) WQCBM3D_vlight(n,k) =
Vlmax_f*(dtanh(Alpha_f*WQCBM3D_ri(k)*4.15/(Vlmax_f*1.0e6)))
                  !
                  !a new formula from Chuck 10/7/99
                  If(UseVtemp_f == 1) WQCBM3D_VTemp(n,k)  = Vtmax_f*(1.0 -
dexp(-b_f*(c(n,k,I_Temp) - tL_f)))* &
                                                                        (1.0 -
dexp(- c_f*(tH_f - c(n,k,I_Temp)))) - voff_f
                        If(WQCBM3D_VTemp(n,k) .lt. 0.0) WQCBM3D_VTemp(n,k)
= 0.0
                        d1 = dzk(k+1)
                   End Do !k
             End If
             !
             d1 = dzk(kt) - z(n)
             settop = 0.0
             !
             If(GenAlgaeModel .and. UseLOTTMethod == 1) Call
ComputeDepthRelatedSolarRadiation

             Do k = kt, k0(i,j)
                   CellVolume = dxx(i,j)*dyy(i,j)*d1
                   temp20 = c(n,k,I_Temp) - 20.0
                   PhytoConc = c(n,k,I_DAP) + c(n,k,I_DFP)
                   If(GenAlgaeModel)Then
                        Do AlgLoop = 1, ngamcs
                              id = I_GAM(AlgLoop)
                              PhytoConc = PhytoConc + c(n,k,id)
```

```
                    End Do !AlgLoop
                End If

                PhytoConc = PhytoConc*anc    !SP 08/31/2011 added anc
                !
                !VSK: 03-06-2010
                If(UseLOTTMethod == 1) Then
                      !
                      !Light limitation for algae growth (I_Phyt)
                      xri_d = WQCBM3D_ri(k)/rins_d*dexp(1.-
WQCBM3D_ri(k)/rins_d)  !light limtation
                      xri_f = WQCBM3D_ri(k)/rins_f*dexp(1.-
WQCBM3D_ri(k)/rins_f)  !light limtation
                Else
                      !!!Note that original 1998 version and
Budd/Deschutes TMDL versions
                      !!!did not convert rins_d and rins_f to same units
as solar radiation
                      !
                      !VSK: 06-24-2010 Commented since unit conversions
are done in WQMReadsubs.f90
                      !Greg originally asked to use this conversions
here.
                      !We then decided to add multiple units for light
                      !But this was not updated
                      !rins_d = rins_d * 4.1868 * 10000.0 / 86400.0
!convert from cal/cm^2/d to W/m^2
                      !rins_f = rins_f * 4.1868 * 10000.0 / 86400.0
!convert from cal/cm^2/d to W/m^2

                      !Steele eqn depth-integrated over thickness of the
layer per QUAL2Kw documentation
                      !Steele eqn for diatoms
               If (rins_d == 0) then
                            xri_d = 1.0
               Else
                            alpha0 = RTopPAR(n,k) / rins_d
                            alpha1 = RTopPAR(n,k) / rins_d *
RExtPAR(n,k)
                            xri_d = Exp(1.0) * (Exp(-alpha1) - Exp(-
alpha0)) / KeCalc(n,k)
               End if
                      !Steele eqn for dinoflagellates
               If (rins_f == 0) then
                            xri_f = 1.0
               Else
                   alpha0 = RTopPAR(n,k) / rins_f
                   alpha1 = RTopPAR(n,k) / rins_f * RExtPAR(n,k)
                            xri_f = Exp(1.0) * (Exp(-alpha1) - Exp(-
alpha0)) / KeCalc(n,k)
               End if
                End If
                !
                !light limitation factor for diatoms in cell n layer k
```

```
                  WQCBM3D_LL(n,k,1) = xri_d
                  !light limitation factor for dinos in cell n layer k
                  WQCBM3D_LL(n,k,2) = xri_f
                  !N limitation factor for diatoms in cell n layer k
                  WQCBM3D_NL(n,k,1) = (c(n,k,I_NH3) + c(n,k,I_NO3))
/(kmn_d + c(n,k,I_NH3) + c(n,k,I_NO3))
                  !N limitation factor for diatoms in cell n layer k
                  WQCBM3D_NL(n,k,2) = (c(n,k,I_NH3) + c(n,k,I_NO3))
/(kmn_f + c(n,k,I_NH3) + c(n,k,I_NO3))
                  !temperature limitation factor for diatoms in cell n
layer k
                  !
                  !Ambrose Bug Fix
                  WQCBM3D_TL(n,k,1) = th1c_d**temp20
                  !temperature limitation factor for diatoms in cell n
layer k
                  !
                  !Ambrose Bug Fix
                  WQCBM3D_TL(n,k,2) = th1c_f**temp20
                  !
                  !Ammonia preference factor for alage growth
                  epilson = 1.0e-30
                  pnh3_d =
c(n,k,I_NH3)*c(n,k,I_NO3)/((kmn_d+c(n,k,I_NH3))*(kmn_d+c(n,k,I_NO3)))&
                                    +
c(n,k,I_NH3)*kmn_d/(epilson+(c(n,k,I_NH3)+c(n,k,I_NO3))*(kmn_d+c(n,k,I_NO
3)))
                  pnh3_f =
c(n,k,I_NH3)*c(n,k,I_NO3)/((kmn_f+c(n,k,I_NH3))*(kmn_f+c(n,k,I_NO3)))&
                      +
c(n,k,I_NH3)*kmn_f/(epilson+(c(n,k,I_NH3)+c(n,k,I_NO3))*(kmn_f+c(n,k,I_NO
3)))

                  option_fnh3 = 0
                  If(option_fnh3.eq.1) Then
                        kinhib = 0.056
                        fnh3 = 1./(1.+c(n,k,I_NH3)/kinhib)
                  End If

                  If(acc(I_DAP) == 1 .and. DAPStatus == 1) Call
WQCBM3DPhytoPlanktonRatesForDiatoms
                  If(acc(I_DFP) == 1 .and. DFPStatus == 1) Call
WQCBM3DPhytoPlanktonRatesForDinos
                  !
                  ! -----------  I_NH3,  Ammonia Nitrogen,     g N/m^3   -
---------------------------------
                  p01a = dr_d*anc*c(n,k,I_DAP)
!(a) algae respiration - Diatoms
                  If(xri_f.ge.0.0) Then
                        p01a =   p01a + dr_f*anc*c(n,k,I_DFP)
!(a) algae respiration - Dinos
                  Else
                        p01a =   p01a - dr_f*anc*pnh3_f*c(n,k,I_DFP)
!(a) algae respiration - Dinos
```

```
                  EndIf
!turn off Dino res. during nittime

                  p01b = k71*th71**temp20*PhytoConc &
!(b) Mineralization
                              /(kmnc + PhytoConc)*(c(n,k,I_ON_D))

                  p01c = - gp_d*anc*pnh3_d*c(n,k,I_DAP)
!(c) Algae growth - Diat
                  !SP 08/31/2011 Added dinos
                  p01c = p01c - gp_f*anc*pnh3_f*c(n,k,I_DFP)
!(c) Algae growth - Dino
                  !Changed to p01a from p01c as p01a is for respiration

                  If(option_fnh3.eq.1) Then
! use inhibition
                        p01c =   - gp_d*anc*c(n,k,I_DAP)
!(c) algae growth - Diat
                        If(xri_f.eq.0.0) Then
!(c) algae growth - Dino
                              p01c =   p01c - dr_f*anc*c(n,k,I_DFP)
!    dark respiration
                        Else
                              p01c =   p01c - gp_f*anc*c(n,k,I_DFP)
                        End If
                  End If

                  p01d =   - k12*th12**temp20*c(n,k,I_DO)  &
!(d) nitrification
                               /(knit + c(n,k,I_DO))*c(n,k,I_NH3)
                  h(n,k,I_NH3) = acc(I_NH3)*(h(n,k,I_NH3)+(p01a + p01b +
p01c + p01d)*CellVolume)

                  NH4Nitrif  = dabs(p01d)

                  !
                  !-----------  I_N03, Nitrate Nitrogen,      g N/m^3  ---
------------------------------
                  p02a =   k12*th12**temp20*c(n,k,I_DO)  &
!(a) nitrification
               /(knit+c(n,k,I_DO))*c(n,k,I_NH3)

                  p02b = - gp_d*anc*(1.0-pnh3_d)*c(n,k,I_DAP)
!(b) algae growth -Diat
                  !SP 08/31/2011 added growth for dinos
                  p02b = p02b - gp_f*anc*(1.0-pnh3_f)*c(n,k,I_DFP)
!(b) algae growth -Diat

                  If(xri_f.lt.0.0) Then
                        p02b =   p02b - dr_f*anc*(1.0-pnh3_f)*c(n,k,I_DFP)
!(a) algae respiration - Dinos
                  Endif
```

```
                p02c = - k2d*th2d**temp20*kno3         &
!(c) denitrification
                           /(kno3 + c(n,k,I_DO))*c(n,k,I_NO3)
                h(n,k,I_NO3) = acc(I_NO3)*(h(n,k,I_NO3) + (p02a + p02b +
p02c)*CellVolume)

                Denitr = dabs(p02c)

                !
                !-----------  I_PO4,  Inorganic Phosphorus, g P/m^3  ---
------------------------------
                p03a =   dr_d*apc*c(n,k,I_DAP)
!(a) respiration - Diatoms
                !GP,VSK 07-30-2007
                !p03a =   p03a + dr_f*apc*c(n,k,I_DFP)
!(a) respiration - Dinos
            If(xri_f.ge.0.0) Then
                 p03a =   p03a + dr_f*apc*c(n,k,I_DFP)     !(a) daytime
respiration of dinos releases PO4
            Else
                p03a =   p03a - dr_f*apc*c(n,k,I_DFP)     !(a) dark
respiration of dinos takes up PO4
            EndIf
                p03b =   k83*th83**temp20*PhytoConc/(kmpc + PhytoConc) &
!(b) mineralization
                           *c(n,k,I_OP_D)
                p03c =   - gp_d*apc*c(n,k,I_DAP)
!(c) algae growth - Diatos
                p03c =   p03c - &
                           gp_f*apc*c(n,k,I_DFP)
!(c) algae growth - Dinos
                h(n,k,I_PO4) = acc(I_PO4)*(h(n,k,I_PO4) + (p03a + p03b +
p03c)*CellVolume)
                !
                ! -----------  I_DAP, Diatoms, g C/m^3  ----------------
----------------
                p41a =   gp_d*c(n,k,I_DAP)
!(a) Algae growth - Diatoms
                p41b = - dr_d*c(n,k,I_DAP)
!(b) Respiration- Diatoms
                p41c = - dd_d*c(n,k,I_DAP)
!(c) Algae death- Diatoms
                p41d = - fe_d*gp_d*c(n,k,I_DAP)
!(d) Excretion- Diatoms
                p41e = - ggraze_d
!(e) Zoopla. graze- Diatoms
                p41f = - vs4_d*c(n,k,I_DAP)/d1 + &
                         vs4_d*c(n,k-1,I_DAP)*SetTop/dzk(k-1)
!(f) Algae settling- Diatoms
                h(n,k,I_DAP) = acc(I_DAP)*DAPStatus*(h(n,k,I_DAP)+(p41a
+ p41b + p41c + p41d + p41e + p41f)*CellVolume)
                !
                !net DAP PProd from growth, respiration, death, and
excretion
```

```
                WQCBM3D_PProd(n,k,1) = p41a + p41b + p41c + p41d
                !
                ! -----------  I_DFP, Dinos, g C/m^3  ------------------
--------------
                p42a =   gp_f*c(n,k,I_DFP)
!(a) Algae growth - Dinos
                !
                !GP,VSK 07-30-2007
                !p42b = - dr_f*c(n,k,I_DFP)
!(b) Respiration- Dinos
                If(xri_f.ge.0.0) Then
                 p42b = - dr_f*c(n,k,I_DFP)    !daytime respiration is a
loss term for Dinos
            Else
                p42b = dr_f*c(n,k,I_DFP)      !dark respiration is a
growth term for Dinos
            EndIf
                 p42c = - dd_f*c(n,k,I_DFP)
!(c) Algae death- Dinos
                p42d = - fe_f*gp_f*c(n,k,I_DFP)
!(d) Excretion- Dinos
                p42e = - ggraze_f
!(e) Zoopla. graze- Dinos
                !
                !Day Time
                If(DFPStatus == 1) Then
                      If (WQCBM3D_ri(k).GT.0) Then
                            If (k == k0(i,j)) Then
                                  p42f = (-vgrav - WQCBM3D_VTemp(n,k) -
WQCBM3D_vlight(n,k))*c(n,k,I_DFP)/d1 + &
                                                vgrav*c(n,k-
1,I_DFP)/dzk(k-1)
                            Else
                                  !
                                  !Ambrose Bug Fix
                                  If (WQCBM3D_ri(k) > rins_f .and.
WQCBM3D_ri(k+1) > rins_f) Then
                                        p42f =  (-WQCBM3D_vlight(n,k)
- WQCBM3D_VTemp(n,k)   - vgrav)*c(n,k,  I_DFP)/d1 + &
                                                   ( WQCBM3D_vlight(n,k-1)
+ WQCBM3D_VTemp(n,k-1) + vgrav)*c(n,k-1,I_DFP)*SetTop/dzk(k-1)
                                  Else If (WQCBM3D_ri(k) > rins_f .and.
WQCBM3D_ri(k+1) <= rins_f) Then
                                        p42f =  ((-WQCBM3D_vlight(n,k)
- WQCBM3D_VTemp(n,k))  - vgrav)*c(n,k,  I_DFP)/d1 + &
                                                      (
WQCBM3D_vlight(n,k-1) + WQCBM3D_VTemp(n,k-1) + vgrav)*c(n,k-
1,I_DFP)*SetTop/dzk(k-1) + &
                                                      (
WQCBM3D_vlight(n,k+1) + WQCBM3D_VTemp(n,k+1)
)*c(n,k+1,I_DFP)/dzk(k+1)
                                  Else If (WQCBM3D_ri(k) <= rins_f .and.
WQCBM3D_ri(k-1) > rins_f) Then
```

```
                                        p42f = (((-WQCBM3D_vlight(n,k)
- WQCBM3D_VTemp(n,k)))*SetTop - vgrav)*c(n,k,I_DFP)/d1 + &
                                                (
WQCBM3D_vlight(n,k-1) + WQCBM3D_VTemp(n,k-1)        + vgrav)*c(n,k-
1,I_DFP)*SetTop/dzk(k-1) + &
                                                (
WQCBM3D_vlight(n,k+1) + WQCBM3D_VTemp(n,k+1)
)*c(n,k+1,I_DFP)/dzk(k+1)
                                  Else If (WQCBM3D_ri(k) <= rins_f .and.
WQCBM3D_ri(k-1) <= rins_f) then
                                        p42f = (((-WQCBM3D_vlight(n,k)
- WQCBM3D_VTemp(n,k) ))*SetTop - vgrav)*c(n,k,I_DFP)/d1 + &
                                                (
WQCBM3D_vlight(n,k+1) + WQCBM3D_VTemp(n,k+1))*c(n,k+1,I_DFP)/dzk(k+1) + &
                                        vgrav*SetTop*c(n,k-
1,I_DFP)/dzk(k-1)
                                  End If
                            End If
                      !
                      !Night time
                      Else
                            If (k == k0(i,j)) Then
                                  p42f = (-vgrav)*c(n,k,I_DFP)/d1 + &
                                     ( vgrav + WQCBM3D_VTemp(n,k-
1))*c(n,k-1,I_DFP)/dzk(k-1)
                            Else
                                  p42f = (-WQCBM3D_VTemp(n,k)  -
vgrav)*c(n,k,I_DFP)/d1 + &
                                        (WQCBM3D_VTemp(n,k-1) +
vgrav)*c(n,k-1,I_DFP)*SetTop/dzk(k-1)
                            End If
                      End If
                End If

                h(n,k,I_DFP) = acc(I_DFP)*DFPStatus*(h(n,k,I_DFP)+(p42a
+ p42b + p42c + p42d + p42e + p42f)*CellVolume)
                !
                !net DFP PProd from growth, respiration, death, and
excretion
                WQCBM3D_PProd(n,k,2) = p42a + p42b + p42c + p42d
                If(mdltime >= 56671560 .and. n == 2376)Then
                !
     Write(7324,'(i4,",",f14.6,",",f14.6,",",f14.6,",",f14.6,",",f14.6,"
,",f14.6)')k, xri_f, gp_f,p42a,p42b,p42c,p42d
                End If
                !
                !VSK 10-27-2005
                PhytoPhoto = abs(p41a) + abs(p42a)
                PhytoResp  = abs(p41b) + abs(p42b)
                PhytoDeath = abs(p41c) + abs(p42c)
                !
                !-----------  I_DO,   Dissolved Oxygen,     mg O2/m^3  -
----------------------------
                !Ambrose Bug Fix
```

```
                If(UseLOTTMethod == 1) Then
                !
                   !use the original LOTT study method
                 p06a =   gp_d*(32./12. + 0.8*32./12.*(1. - pnh3_d) &
!(a) Photo. - Diat
                          + 0.2*32./12.*(pnh3_d))*c(n,k,I_DAP)
                 p06a =   p06a + &
                        gp_f*(32./12. + 0.8*32./12.*(1. - pnh3_f) &
!(a) Photo. - Dinos
                         + 0.2*32./12.*(pnh3_f))*c(n,k,I_DFP)
              Else
                     !
                 !use the WASP EUTRO method adapted from WASP
DISSOXYG.FOR
                 !note: c(n,k,I_DAP) and c(n,k,I_DFP) is phyto biomass
in mgC/L = gC/m^3
                 !growth of diatoms (I_DAP) using CO2 and NH3
                 p06a =              pnh3_d  * gp_d * c(n,k,I_DAP) * 32.
/ 12.
                 !growth of diatoms (I_DAP) using CO2 and NO3 (2NO3 =
2NH3 + 3O2)
                 p06a = p06a + (1. - pnh3_d) * gp_d * c(n,k,I_DAP) * 32.
* (1./12. + 1.5 * anc / 14.)
                 !growth of dinoflagellates (I_DFP) using CO2 and NH3
                   p06a = p06a +       pnh3_f  * gp_f * c(n,k,I_DFP) *
32. / 12.
                    !growth of dinoflagellates (I_DFP) using CO2 and NO3
(2NO3 = 2NH3 + 3O2)
                     p06a = p06a + (1. - pnh3_f) * gp_f * c(n,k,I_DFP)
* 32. * (1./12. + 1.5 * anc / 14.)
                     !
                 !! here is the equivalent code from WASP DISSOXYG.FOR:
                 !C        Growth of phytoplankton using CO2 and NH3
                 !             SR19PA = SR19PA +
PNH3G1(i)*GP1(i)*PHYT(i)*32./12.           !JLM 5/07 Multi-Algae
Additions
                 !C        Growth of phytoplankton using CO2 and NO3
(2NO3 = 2NH3 + 3O2)    !JLM 5/07 Multi-Algae Additions
                 !             SR19PB = SR19PB + (1. -
PNH3G1(i))*GP1(i)*PHYT(i)            !JLM 5/07 Multi-Algae Additions
                 !     .        *32.*(1./12. + 1.5*NCRB(i)/14.)
!JLM 5/07 Multi-Algae Additions
              End If

                 p06b =   - 32./12.*dr_d*c(n,k,I_DAP)
!(e) Respiration -Diat
                 p06b =   p06b &
                          - 32./12.*dr_f*c(n,k,I_DFP)
!(e) Respiration -Dino
                 WQCBM3D_PhytoProd(n,k) = (p06a +
p06b)*CellVolume*86400.0           !gm/day

                 p06c = -(kdf*thd**temp20*c(n,k,I_CBOD_F) + &
```

```
                          kds*thd**temp20*c(n,k,I_CBOD_S))* &
!Heterotropic
                             c(n,k,I_DO)/(kbod + c(n,k,I_DO))
!Respiration


                  p06d = - 64./14.*k12*th12**temp20*c(n,k,I_DO)  &
!(c) Nitrification
                          /(knit+c(n,k,I_DO))*c(n,k,I_NH3)

                  h(n,k,I_DO)  = acc(I_DO)*(h(n,k,I_DO) + (p06a + p06b +
p06c + p06d)*CellVolume)
                  !
                  ! -----------  I_CBOD_F, Fast Reacting Carbonaceous BOD,
g O2/m^3  -------------------
                  p15a =   aoc*foc*fd5*dd_d*c(n,k,I_DAP)
!(a) Algae death - Diat
                  p15a =   p15a + &
                           aoc*foc*fd5*dd_f*c(n,k,I_DFP)
!(a) Algae death - Dinos
                  !SP 08/31/2011
                  p15a =     p15a !+ aoc*foc*dr_d*c(n,k,I_DAP)
                   !(a) Algae respiration - Diat
                  If(xri_f.ge.0.0) Then
                   p15a = p15a !+ aoc*foc*dr_f*c(n,k,I_DFP)    !daytime
respiration is a loss term for Dinos
             Else
                 p15a = p15a !- aoc*foc*dr_f*c(n,k,I_DFP)      !dark
respiration is a growth term for Dinos
             EndIf
                  !End SP 08/31/2011

                  p15b =   aoc*fe_d*gp_d*c(n,k,I_DAP)
!(b) Algae excretion- Diat
                  p15b =   p15b + &
                           aoc*fe_f*gp_f*c(n,k,I_DFP)
!(b) Algae excretion - Dinos
                  p15d = - kdf*thd**temp20*c(n,k,I_DO) &
!(d) Heterotrophic
                             /(kbod + c(n,k,I_DO))*c(n,k,I_CBOD_F)
!    respiration
                  p15e = - 5./4.*32./14.*k2d*th2d**temp20*kno3  &
!(e) Denitrification
                             /(kno3 + c(n,k,I_DO))*c(n,k,I_NO3)
                  p15c = 0.0
                  h(n,k,I_CBOD_F) =
acc(I_CBOD_F)*(h(n,k,I_CBOD_F)+(p15a*(1-PhytoDandEinWQADD) + p15b*(1-
PhytoDandEinWQADD) + p15c + p15d + p15e)*CellVolume)

                  !
                  !-----------  I_CBOD_S, Slow Reacting Carbonaceous BOD,
g O2/m^3   ------------------
                  p16a =   aoc*foc*(1 - fd5)*dd_d*c(n,k,I_DAP)
!(a) Algae death - Diat
```

```
                p16a =   p16a + &
                         aoc*foc*(1 - fd5)*dd_f*c(n,k,I_DFP)
!(a) Algae death - Dinos
                !
                !Hydrolysis
                !Ambrose Bug Fix
           p16c =   aoc * kpd9f*thpd9f**temp20*c(n,k,I_OC_P_F) + &
                    aoc * kpd9s*thpd9s**temp20*c(n,k,I_OC_P_S)

                p16d = - kds*thd**temp20*c(n,k,I_DO) &
!(d) Oxidation
                            /(kbod + c(n,k,I_DO))*c(n,k,I_CBOD_S)
                p16e = - 5./4.*32./14.*k2d*th2d**temp20*kno3  &
!(e) Denitrification
                            /(kno3 + c(n,k,I_DO))*c(n,k,I_NO3)
                h(n,k,I_CBOD_S)=
acc(I_CBOD_S)*(h(n,k,I_CBOD_S)+(p16a*(1-PhytoDandEinWQADD) + p16c + p16d
+ p16e)*CellVolume)

                CBODDecay      = abs(p15d) + abs(p16d)
                CBODOxidation  = abs(p16d)

                !If(n == 2 .and. k ==
kt)Write(7323,'(f14.6,",",f14.6,",",f14.6,",",f14.6)')p15d*CellVolume,
p16d*CellVolume,PhytoResp*CellVolume, PhytoPhoto*CellVolume
                !If(n == 2 .and. k == kt)Write(7325,*)mdltime/(24*60)+1
                !
                !-----------  I_ON_D, Dissolved Organic Nitrogen, g
N/m^3       --------------------
                p10a =   dd_d*anc*fon*c(n,k,I_DAP)
!(a) Algae death - Diat
                p10a =   p10a + &
                         dd_f*anc*fon*c(n,k,I_DFP)
!(a) Algae death - Dino
                p10b =   anc*fe_d*gp_d*c(n,k,I_DAP)
!(b) Algae excretion
                p10b =   p10b + &
                         anc*fe_f*gp_f*c(n,k,I_DFP)
!(b) Algae excretion
                p10c = - k71*th71**temp20*PhytoConc &
!(c) Mineralization
                           /(kmnc + PhytoConc)*c(n,k,I_ON_D)
                p10d =   kh7p*thh7p**temp20*c(n,k,I_ON_P)
!(d) Hydrolysis
                h(n,k,I_ON_D) = acc(I_ON_D)*(h(n,k,I_ON_D) + (p10a +
p10b + p10c + p10d)*CellVolume)
           !
                !-----------  I_ON_P, Particulate Organic Nitrogen,
g N/m^3     ---------------
                p11a =   dd_d*anc*(1 - fon)*c(n,k,I_DAP)
!(a) Algae death- Diat
                p11a =   p11a + &
                         dd_f*anc*(1 - fon)*c(n,k,I_DFP)
!(a) Algae death - Dinos
```

```
                  p11b =   anc*ggraze_d
!(b) Micrograzing
                  p11b =   p11b + &
                           anc*ggraze_f
!(b) Micrograzing
                  p11c = - kh7p*thh7p**temp20*c(n,k,I_ON_P)
!(c) Hydrolysis
                  p11d = - vs7*c(n,k,I_ON_P)/d1 + &
                           vs7*c(n,k-1,I_ON_P)*SetTop/dzk(k-1)
                  h(n,k,I_ON_P) =
acc(I_ON_P)*(h(n,k,I_ON_P)+(p11a+p11b+p11c+p11d)*CellVolume)
                  OrgNHydr   = abs(p10d) + abs(p11c)
                  !
                  !-----------  I_OP_D,  Dissovled Organic Phosphorus,   g
P/m^3       -------------
                  p13a =   dd_d*apc*fop*c(n,k,I_DAP)
!(a) Algae death- Diat
                  p13a =   p13a + &
                           dd_f*apc*fop*c(n,k,I_DFP)
!(a) Algae death- Dino
                  p13b =   apc*fe_d*gp_d*c(n,k,I_DAP)
!(b) Algae excretion
                  p13b =   p13b + &
                           apc*fe_f*gp_f*c(n,k,I_DFP)
!(b) Algae excretion
                  p13c = - k83*th83**temp20*PhytoConc &
!(c) Mineralization
                             /(kmpc + PhytoConc)*c(n,k,I_OP_D)
                  p13d =   kh8p*thh8p**temp20*c(n,k,I_OP_P)
!(d) Hydrolysis
                  h(n,k,I_OP_D) = acc(I_OP_D)*(h(n,k,I_OP_D)+(p13a + p13b
+ p13c + p13d)*CellVolume)
                  !
                  !-----------  I_OP_P, Particulate Organic Phosphorus,
g P/m^3  -----------------
                  p14a =   dd_d*apc*(1 - fop)*c(n,k,I_DAP)
!(a) Algae death
                  p14a =   p14a + &
                           dd_f*apc*(1 - fop)*c(n,k,I_DFP)
!(a) Algae death
                  p14b =          apc*ggraze_d
!(b) Micrograzing
                  p14b =   p14b + apc*ggraze_f
!(b) Micrograzing
                  p14c = - kh8p*thh8p**temp20*c(n,k,I_OP_P)
!(c) Hydrolysis
                  p14d = - vs8*c(n,k,I_OP_P)/d1 + &
                           vs8*c(n,k-1,I_OP_P)*SetTop/dzk(k-1)
!(d) Settling
                  !p14e = - k83*th83**temp20*c(n,k,I_Phyt) &
!(c) mineralization (*)
                  !          /(kmpc+c(n,k,I_Phyt))*c(n,k,I_OP_P)
                  p14e = 0.0
```

```
                h(n,k,I_OP_P) = acc(I_OP_P)*(h(n,k,I_OP_P)+(p14a + p14b
+ p14c + p14d + p14e)*CellVolume)
                OrgPHydr   =  abs(p13d) + abs(p14c)
                !
                !-----------  I_OC_P_F, Fast Reacting Particulate
Organic Carbon, g C/m^3   -----------
                !SP 05/16/2008
                p17a =   dd_d*fd9f*(1 - foc)*c(n,k,I_DAP)
!(a) Algae death
                p17a =   p17a + &
                         dd_f*fd9f*(1 - foc)*c(n,k,I_DFP)
!(a) Algae death
                p17b =   fg9f*(ggraze_d + ggraze_f)                 !(b)
Micrograzing
                !End SP 05/16/2008
                p17c = - kpd9f*thpd9f**temp20*c(n,k,I_OC_P_F)
!(c) Hydrolysis
                p17d = - vs9*c(n,k,I_OC_P_F)/d1 + &
                         vs9*c(n,k-1,I_OC_P_F)*SetTop/dzk(k-1)
!(d) Settling
                h(n,k,I_OC_P_F) =
acc(I_OC_P_F)*(h(n,k,I_OC_P_F)+(p17a*(1-PhytoDandEinWQADD) + p17b + p17c
+ p17d)*CellVolume)
                !
                !-----------  I_OC_P_S, Slow Reacting Particulate
Organic Carbon, g C/m^3   -----------
                !SP 05/16/2008
                p18a =   dd_d*fd9s*(1 - foc)*c(n,k,I_DAP)
!(a) Algae death
                p18a =   p18a + &
                         dd_f*fd9s*(1 - foc)*c(n,k,I_DFP)
!(a) Algae death
                p18b =   fg9s*(ggraze_d + ggraze_f)               !(b)
Micrograzing
                !End SP 05/16/2008
                p18c = - kpd9s*thpd9s**temp20*c(n,k,I_OC_P_S)
!(c) Hydrolysis
                p18d = - vs9*c(n,k,I_OC_P_S)/d1 + &
                         vs9*c(n,k-1,I_OC_P_S)*SetTop/dzk(k-1)
!(d) Settling
                h(n,k,I_OC_P_S) =
acc(I_OC_P_S)*(h(n,k,I_OC_P_S)+(p18a*(1-PhytoDandEinWQADD) + p18b + p18c
+ p18d)*CellVolume)
                !
                !-----------  I_OC_P_R, Refractory Reacting Particulate
Organic Carbon, g C/m^3  ------
                !SP 05/16/2008
                p19a =   dd_d*fd9r*(1 - foc)*c(n,k,I_DAP)
!(a) Algae death
                p19a =   p19a + &
                         dd_f*fd9r*(1 - foc)*c(n,k,I_DFP)
!(a) Algae death
```

```
                  p19b =   fg9r*(ggraze_d + ggraze_f)              !(b)
Micrograzing
                  !End SP 05/16/2008
                  p19c = - vs9*c(n,k,  I_OC_P_R)/d1 + &
                              vs9*c(n,k-1,I_OC_P_R)*SetTOP/dzk(k-1)
!(c) Settling
                  h(n,k,I_OC_P_R) = acc(I_OC_P_R)*(h(n,k,I_OC_P_R) +
(p19a*(1-PhytoDandEinWQADD) + p19b + p19c)*CellVolume)

                  !If(n == 2 .and. k ==
kt)Write(7324,'(f14.6,",",f14.6,",",f14.6,",",f14.6,",",f14.6,",",f14.6,"
,")')mdltime/(24*60)+1,h(n,k,I_CBOD_F), h(n,k,I_CBOD_S), h(n,k,I_OC_P_F),
h(n,k,I_OC_P_S), h(n,k,I_OC_P_R)

                  If(GenAlgaeModel .and. SetGAMSSTermsInsideWQM == 1) Call
GAMSSTermsInsideWQM
                  !
                  !
                  aoc1 = aoc
                  !SP 05/16/2008
                  foc_WQADD = foc;       fd5_WQADD = fd5;       fd9f_WQADD
= fd9f;           fd9s_WQADD = fd9s;
                  fd9r_WQADD = fd9r;           fon_WQADD = fon;
      fop_WQADD = fop
                  !End SP 05/16/2008
                  If(AddWaterQuality .and. iwqaddcss == 1) Call
WQADD3DSSTerms

                  settop = 1.0            ! reassign settling for the top
layer, before next k loop
                  d1 = dzk(k+1)           ! update layer thickness for
next k loop
                  !Print*, n, k
            End Do ! end of k (for layer)


            If(WriteWQMOutputVariables == 1) Call
ComputeWQCBMInternalOutputVariables


      End Do  ! end of n (for each cell)

      !Write(7323,'(f14.6,",",f14.6,",",f14.6,",",f14.6,",",f14.6,",",f14
.6,",",f14.6)')mdltime,mdltime/(24*60)+1,WQCBM3D_PPROD(2376,ktwb(1),2)*86
400.0,WQCBM3D_PPROD(2376,k0(76,108),2)*86400.0

      If(mdltime > 51109022)Then
            Continue
      End If

Return

      Entry ComputeReAerationCoefficientInWQCBM
```

```
            TempWad10 = wad/((2.0/10.0)**(1.0/7.0))

            Select Case(SDOERMethod)
                  !
                  !Wanninkohf et al. 1991
                  Case(1)
                        ! (1) Change wad from 2 meter speed to 10 meter
speed for
                        !     reaeration coefficient 1/7th power law, vsk
                        ! (2) Use Wanninkohf et al. (1991) model (cm/hr)
                        ! (3) Convert k2_wind from cm/hr to m/s and add
factors
                        k2_wind = 0.464*((TempWad10)**1.61)*((1447.*dexp(-
0.0537*c(n,kt,I_Temp)))/600.)**0.5
                        k2 = k2_wind/(100.*3600.0)/1.   !+k2_vel
                        k2T = k2*Thtk2**(c(n,kt,I_Temp) - 20.0)
                        WQk2T(n) = k2T
                  !
                  !Chen & Kanwisher 1963
                  Case(2)
                        If(TempWad10 > 10.0) TempWad10 = 10.0
                        DMO2     = 2.4d-9      !Unit is m2/s
                        k2_wind  = (200.d0 - 60.d0*dsqrt(TempWad10))*1.0d-
6
                        k2       = DMO2/k2_wind      !Unit is m/s
                        k2       = k2*86400.d0 !Unit is m/d
                        If(ApplyMinReAeration .and. k2 <
MinReAerationCoefficient)k2 = MinReAerationCoefficient
                        k2       = k2/86400.d0 !Unit again is m/d
                        k2T      = k2*Thtk2**(c(n,kt,I_Temp) - 20.0)
                        WQk2T(n) = k2T
                  Case Default
                        ! (1) Change wad from 2 meter speed to 10 meter
speed for
                        !     reaeration coefficient 1/7th power law, vsk
                        ! (2) Use Wanninkohf et al. (1991) model (cm/hr)
                        ! (3) Convert k2_wind from cm/hr to m/s and add
factors
                        k2_wind = 0.464*((TempWad10)**1.61)*((1447.*dexp(-
0.0537*c(n,kt,I_Temp)))/600.)**0.5
                        k2 = k2_wind/(100.*3600.0)/1.   !+k2_vel
                        k2T = k2*Thtk2**(c(n,kt,I_Temp) - 20.0)
                        WQk2T(n) = k2T
            End Select

      Return



      Entry ComputeWQCBMInternalOutputVariables

            d1 = el(kt) - z(n)
            Sum = 0.0d+00
            Do k = kt, k0(i,j)
```

```
                  Sum = Sum + (c(m1(i,j),k,I_DAP)/WQCBM3D_c2chla_d + &
                                    c(m1(i,j),k,I_DFP)/WQCBM3D_c2chla_f)*d1

                  d1 = dzk(k+1)
            End Do

            BioMass(n) = Sum

      Return


      Entry WQCBMSetConstantSedimentFluxes
            !
            !SOD Use values from rates and constants
            sod = sodm*thsod**(c(n,k0(i,j),I_Temp) - 11.0)
            h(n,k0(i,j),I_DO) = acc(I_DO)*(h(n,k0(i,j),I_DO) +
sod*CellArea)
            !
            !NH3
            h(n,k0(i,j),I_NH3) = acc(I_NH3)*(h(n,k0(i,j),I_NH3) +
sednh3m*CellArea)
            !
            !NO3
            h(n,k0(i,j),I_NO3) = acc(I_NO3)*(h(n,k0(i,j),I_NO3) +
sedno3m*CellArea)
            !
            !PO4
            h(n,k0(i,j),I_PO4) = acc(I_PO4)*(h(n,k0(i,j),I_PO4) +
sedpo4m*CellArea)

      Return


      ENTRY WQCBM3DPhytoPlanktonRatesForDiatoms
            !
            !Zooplankton Grazing
            Select Case(ZPGMode_d)
                  !
                  !g/m^3-day
                  !Ambrose Bug Fix
                  Case(ConstantGrazing)
                        ggraze_d = kgmicro_d*thkt_d**temp20 +
kgmacro_d*thkt_d**temp20                !
                  !1/day
                  !grazing function by Boatman (10/17/97)
                  Case(LinearGrazing)
                        gmicro_d    = kgmicro_d*thkt_d**temp20*c(n,k,I_DAP)
                        gmacro_d    = kgmacro_d*thkt_d**temp20*c(n,k,I_DAP)
                        ggraze_d    = gmicro_d + gmacro_d
                  !
                  !(m^3/g-day)
                  !density dependent
                  Case(DensityDependentGrazing)
```

```
                        gmicro_d    =
kgmicro_d*c(n,k,I_DAP)*thkt_d**temp20*c(n,k,I_DAP)
                        gmacro_d    =
kgmacro_d*c(n,k,I_DAP)*thkt_d**temp20*c(n,k,I_DAP)
                        ggraze_d    = gmicro_d + gmacro_d
            End Select
            !
            !Phytolankton growth  - (I_Phyt; C4)
            !plimit = phosphorus limiting (set 1.0 for estuaries)
            plimit      = acc(I_PO4)*(c(n,k,I_PO4)/(kmp_d + c(n,k,I_PO4)))
            If(plimit == 0.0 .and. plc == 0.0) Then
                  plimit = 1.0
            Else
                  plimit = plimit**(plc)
            End If
            !
            !P limitation factor for diatoms in cell n layer k
            WQCBM3D_PL(n,k,1) = plimit

            gp_d  = k1c_d*th1c_d**temp20*xri_d*dmin1((c(n,k,I_NH3) +
c(n,k,I_NO3))    &
                        /(kmn_d + c(n,k,I_NH3) +
c(n,k,I_NO3)),(plimit**acc(I_PO4)))
            !
            !Respiration
            dr_d  = k1r_d*th1r_d**temp20
            !
            !Death
            !dd_d = k1d_d
            !
            !Death; GUI Update
             dd_d = k1d_d*th1d_d**temp20 !SP 11/14/2011 Change to theta
for death


      Return


      ENTRY WQCBM3DPhytoPlanktonRatesForDinos
            !
            !Zooplankton Grazing
            Select Case(ZPGMode_f)
                  !
                  !g/m^3-day
                  !Ambrose Bug Fix
                  Case(ConstantGrazing)
                        ggraze_f = kgmicro_f*thkt_f**temp20 +
kgmacro_f*thkt_f**temp20
                  !
                  !1/day
                  !grazing function by Boatman (10/17/97)
                  Case(LinearGrazing)
                        !SP 05/16/2008
                        gmicro_f    = kgmicro_f*thkt_f**temp20*c(n,k,I_DFP)
```

```
                        gmacro_f    = kgmacro_f*thkt_f**temp20*c(n,k,I_DFP)
                        !End SP 05/16/2008
                        ggraze_f    = gmicro_f + gmacro_f
                  !
                  !(m^3/g-day)
                  !density dependent
                  Case(DensityDependentGrazing)
                        gmicro_f    =
kgmicro_f*c(n,k,I_DFP)*thkt_f**temp20*c(n,k,I_DFP)
                        gmacro_f    =
kgmacro_f*c(n,k,I_DFP)*thkt_f**temp20*c(n,k,I_DFP)
                        ggraze_f    = gmicro_f + gmacro_f
            End Select
            !
            !Phytolankton growth  - (I_Phyt; C4)
            !plimit = phosphorus limiting (set 1.0 for estuaries)
            plimit      = acc(I_PO4)*(c(n,k,I_PO4)/(kmp_f + c(n,k,I_PO4)))
            If(plimit == 0.0 .and. plc == 0.0) Then
                  plimit = 1.0
            Else
                  plimit  = plimit**(plc)
            End If

            !P limitation factor for dinos in cell n layer k
            WQCBM3D_PL(n,k,2) = plimit

            gp_f  = k1c_f*th1c_f**temp20*xri_f*dmin1((c(n,k,I_NH3) +
c(n,k,I_NO3))    &
                        /(kmn_f + c(n,k,I_NH3) +
c(n,k,I_NO3)),(plimit**acc(I_PO4)))
            !
            !Respiration
            dr_f        = k1r_f*th1r_f**temp20
            !
            !Death: GUI Change
            !dd_f       = k1d_f
            dd_f        = k1d_f*th1d_f**temp20 !SP 11/14/2011 Change to
theta for death

      Return

      Entry SetWQCBM3DRatesAndConstantsLocalDefault
            !
            !General Variables
            Ke_a         =   WQCBM3D_Ke_a
            Ke_b         =   WQCBM3D_Ke_b
            Ke_c         =   WQCBM3D_Ke_c
            !
            !NH3
            anc               =     WQCBM3D_anc
            k71               =     WQCBM3D_k71
            th71        =     WQCBM3D_th71
            k12               =     WQCBM3D_k12
            th12        =     WQCBM3D_th12
```

```
knit        =     WQCBM3D_knit
sednh3m           =     WQCBM3D_sednh3m
kmnc         =   WQCBM3D_kmnc
!
!NO3
k2d               =     WQCBM3D_k2d
th2d        =     WQCBM3D_th2d
kno3        =     WQCBM3D_kno3
!
!PO4
apc               =     WQCBM3D_apc
k83               =     WQCBM3D_k83
th83        =     WQCBM3D_th83
kmpc        =   WQCBM3D_kmpc
plc               =     WQCBM3D_plc
!
!Diatoms
c2chla_d    =     WQCBM3D_c2chla_d
rins_d            =     WQCBM3D_rins_d
kmn_d       =     WQCBM3D_kmn_d
ZPGMode_d    =   WQCBM3D_ZPGMode_d
kgmicro_d   =     WQCBM3D_kgmicro_d
kgmacro_d   =     WQCBM3D_kgmacro_d
thkt_d            =     WQCBM3D_thkt_d
k1d_d       =     WQCBM3D_k1d_d
k1c_d       =     WQCBM3D_k1c_d
th1c_d            =     WQCBM3D_th1c_d
kmp_d       =     WQCBM3D_kmp_d
k1r_d       =     WQCBM3D_k1r_d
th1r_d            =     WQCBM3D_th1r_d
vs4_d       =     WQCBM3D_vs4_d
fe_d        =     WQCBM3D_fe_d
as_d        =     WQCBM3D_as_d
DAPStatus    =   WQCBM3D_DAPStatus
th1d_d            =     WQCBM3D_th1d_d
!
!Dynoflagellates
c2chla_f    =     WQCBM3D_c2chla_f
rins_f            =     WQCBM3D_rins_f
kmn_f       =     WQCBM3D_kmn_f
ZPGMode_f    =   WQCBM3D_ZPGMode_f
kgmicro_f   =     WQCBM3D_kgmicro_f
kgmacro_f   =     WQCBM3D_kgmacro_f
thkt_f            =     WQCBM3D_thkt_f
k1d_f       =     WQCBM3D_k1d_f
k1c_f       =     WQCBM3D_k1c_f
th1c_f            =     WQCBM3D_th1c_f
kmp_f             =     WQCBM3D_kmp_f
k1r_f       =     WQCBM3D_k1r_f
th1r_f            =     WQCBM3D_th1r_f
vs4_f       =     WQCBM3D_vs4_f
fe_f        =     WQCBM3D_fe_f
as_f        =     WQCBM3D_as_f
DFPStatus    =   WQCBM3D_DFPStatus
```

```
th1d_f           =     WQCBM3D_th1d_f

UseVtemp_f  =   WQCBM3D_UseVtemp_f
Vtmax_f          =     WQCBM3D_Vtmax_f
b_f         =    WQCBM3D_b_f
c_f         =    WQCBM3D_c_f
tL_f        =    WQCBM3D_tL_f
tH_f        =    WQCBM3D_tH_f
Voff_f           =     WQCBM3D_Voff_f
UseVlight_f =    WQCBM3D_UseVLight_f
Vlmax_f          =     WQCBM3D_Vlmax_f
Alpha_f          =     WQCBM3D_Alpha_f
!
!DO
SDOERMethod =   WQCBM3D_SDOERMethod
kdf              =     WQCBM3D_kdf
kds              =     WQCBM3D_kds
thsod       =     WQCBM3D_thsod
sodm        =     WQCBM3D_sodm
sp4              =     WQCBM3D_sp4
sp5              =     WQCBM3D_sp5
sp7         =   WQCBM3D_sp7
ReaerationFactor = WQCBM3D_ReaerationFactor
Thtk2       =     WQCBM3D_Thtk2
!
!CBOD_F
aoc              =     WQCBM3D_aoc
thd              =     WQCBM3D_thd
kbod        =     WQCBM3D_kbod
foc              =     WQCBM3D_foc

r_CBODP    =   WQCBM3D_r_CBODP
r_CBODC    =   WQCBM3D_r_CBODC
r_CBODN    =   WQCBM3D_r_CBODN

!
!CBOD_S
fd5              =     WQCBM3D_fd5
!
!ON_D
kh7p        =     WQCBM3D_kh7p
thh7p       =     WQCBM3D_thh7p
fon              =     WQCBM3D_fon
!
!ON_P
vs7              =     WQCBM3D_vs7
ancp        =     WQCBM3D_ancp
!
!OP_D
kh8p        =     WQCBM3D_kh8p
thh8p       =     WQCBM3D_thh8p
fop              =     WQCBM3D_fop
!
!OP_P
```

```
      vs8               =     WQCBM3D_vs8
      apcp        =     WQCBM3D_apcp
      !
      !OC_P_F

      fd9f        =     WQCBM3D_fd9f
      fg9f        =     WQCBM3D_fg9f
      kpd9f       =     WQCBM3D_kpd9f
      thpd9f            =     WQCBM3D_thpd9f
      vs9               =     WQCBM3D_vs9
      !
      !OC_P_S
      fd9s        =     WQCBM3D_fd9s
      fg9s        =     WQCBM3D_fg9s
      kpd9s       =     WQCBM3D_kpd9s
      thpd9s       =   WQCBM3D_thpd9s
      !
      !OC_P_R
      fd9r        =     WQCBM3D_fd9r
      fg9r        =     WQCBM3D_fg9r

Return

Entry  SetWQCBM3DRatesAndConstantsLocalChange

      RegionNum = WQCBM3DRegionStatus(i,j)
      If(RegionNum == 0) Return
      !
      !General Variables
      Ke_a         =   WQCBM3DA_Ke_a(RegionNum)
      Ke_b         =   WQCBM3DA_Ke_b(RegionNum)
      Ke_c         =   WQCBM3DA_Ke_c(RegionNum)
      !
      !NH3
      anc               =     WQCBM3DA_anc(RegionNum)
      k71               =     WQCBM3DA_k71(RegionNum)
      th71        =     WQCBM3DA_th71(RegionNum)
      k12               =     WQCBM3DA_k12(RegionNum)
      th12        =     WQCBM3DA_th12(RegionNum)
      knit        =     WQCBM3DA_knit(RegionNum)
      sednh3m           =     WQCBM3DA_sednh3m(RegionNum)

      kmnc         =   WQCBM3DA_kmnc(RegionNum)
      !
      !NO3
      k2d         =     WQCBM3DA_k2d(RegionNum)
      th2d  =     WQCBM3DA_th2d(RegionNum)
      kno3  =     WQCBM3DA_kno3(RegionNum)
      !
      !PO4
      apc         =     WQCBM3DA_apc(RegionNum)
      k83         =     WQCBM3DA_k83(RegionNum)
      th83  =     WQCBM3DA_th83(RegionNum)
      kmpc        =   WQCBM3DA_kmpc(RegionNum)
```

```
plc         =     WQCBM3DA_plc(RegionNum)
!
!Diatoms
c2chla_d    =     WQCBM3DA_c2chla_d(RegionNum)
rins_d              =     WQCBM3DA_rins_d(RegionNum)
kmn_d       =     WQCBM3DA_kmn_d(RegionNum)
ZPGMode_d    =   WQCBM3DA_ZPGMode_d(RegionNum)
kgmicro_d   =     WQCBM3DA_kgmicro_d(RegionNum)
kgmacro_d   =     WQCBM3DA_kgmacro_d(RegionNum)
thkt_d              =     WQCBM3DA_thkt_d(RegionNum)
k1d_d       =     WQCBM3DA_k1d_d(RegionNum)
k1c_d       =     WQCBM3DA_k1c_d(RegionNum)
th1c_d              =     WQCBM3DA_th1c_d(RegionNum)
kmp_d       =     WQCBM3DA_kmp_d(RegionNum)
k1r_d       =     WQCBM3DA_k1r_d(RegionNum)
th1r_d              =     WQCBM3DA_th1r_d(RegionNum)
vs4_d       =     WQCBM3DA_vs4_d(RegionNum)
fe_d        =     WQCBM3DA_fe_d(RegionNum)
as_d        =     WQCBM3DA_as_d(RegionNum)
DAPStatus    =   WQCBM3DA_DAPStatus(RegionNum)
th1d_d       =   WQCBM3DA_th1d_d(RegionNum)
!
!Dynoflagellates
c2chla_f    =     WQCBM3DA_c2chla_f(RegionNum)
rins_f              =     WQCBM3DA_rins_f(RegionNum)
kmn_f       =     WQCBM3DA_kmn_f(RegionNum)
ZPGMode_f    =   WQCBM3DA_ZPGMode_f(RegionNum)
kgmicro_f   =     WQCBM3DA_kgmicro_f(RegionNum)
kgmacro_f   =     WQCBM3DA_kgmacro_f(RegionNum)
thkt_f              =     WQCBM3DA_thkt_f(RegionNum)
k1d_f       =     WQCBM3DA_k1d_f(RegionNum)
k1c_f       =     WQCBM3DA_k1c_f(RegionNum)
th1c_f              =     WQCBM3DA_th1c_f(RegionNum)
kmp_f               =     WQCBM3DA_kmp_f(RegionNum)
k1r_f       =     WQCBM3DA_k1r_f(RegionNum)
th1r_f              =     WQCBM3DA_th1r_f(RegionNum)
vs4_f       =     WQCBM3DA_vs4_f(RegionNum)
fe_f        =     WQCBM3DA_fe_f(RegionNum)
as_f        =     WQCBM3DA_as_f(RegionNum)
DFPStatus    =   WQCBM3DA_DFPStatus(RegionNum)
th1d_f              =     WQCBM3DA_th1d_f(RegionNum)
!
!DO
SDOERMethod =   WQCBM3DA_SDOERMethod(RegionNum)
kdf                 =     WQCBM3DA_kdf(RegionNum)
kds                 =     WQCBM3DA_kds(RegionNum)
thsod       =     WQCBM3DA_thsod(RegionNum)
sodm        =     WQCBM3DA_sodm(RegionNum)
sp4                 =     WQCBM3DA_sp4(RegionNum)
sp5                 =     WQCBM3DA_sp5(RegionNum)
sp7          =   WQCBM3DA_sp7(RegionNum)
ReaerationFactor = WQCBM3DA_ReaerationFactor(RegionNum)
Thtk2       =   WQCBM3DA_Thtk2(RegionNum)
!
```

```
      !CBOD_F
      aoc              =     WQCBM3DA_aoc(RegionNum)
      thd              =     WQCBM3DA_thd(RegionNum)
      kbod        =     WQCBM3DA_kbod(RegionNum)
      foc              =     WQCBM3DA_foc(RegionNum)
      r_CBODP     =   WQCBM3DA_r_CBODP(RegionNum)
      r_CBODC     =   WQCBM3DA_r_CBODC(RegionNum)
      r_CBODN     =   WQCBM3DA_r_CBODN(RegionNum)
      !
      !CBOD_S
      fd5              =     WQCBM3DA_fd5(RegionNum)
      !
      !ON_D
      kh7p        =     WQCBM3DA_kh7p(RegionNum)
      thh7p       =     WQCBM3DA_thh7p(RegionNum)
      fon              =     WQCBM3DA_fon(RegionNum)
      !
      !ON_P
      vs7              =     WQCBM3DA_vs7(RegionNum)
      ancp        =     WQCBM3DA_ancp(RegionNum)
      !
      !OP_D
      kh8p        =     WQCBM3DA_kh8p(RegionNum)
      thh8p       =     WQCBM3DA_thh8p(RegionNum)
      fop              =     WQCBM3DA_fop(RegionNum)
      !
      !OP_P
      vs8              =     WQCBM3DA_vs8(RegionNum)
      apcp        =     WQCBM3DA_apcp(RegionNum)
      !
      !OC_P_F

      fd9f        =     WQCBM3DA_fd9f(RegionNum)
      fg9f        =     WQCBM3DA_fg9f(RegionNum)
      kpd9f       =     WQCBM3DA_kpd9f(RegionNum)
      thpd9f           =     WQCBM3DA_thpd9f(RegionNum)
      vs9              =     WQCBM3DA_vs9(RegionNum)
      !
      !OC_P_S
      fd9s        =     WQCBM3DA_fd9s(RegionNum)
      fg9s        =     WQCBM3DA_fg9s(RegionNum)
      kpd9s       =     WQCBM3DA_kpd9s(RegionNum)
      thpd9s       =   WQCBM3DA_thpd9s(RegionNum)
      !
      !OC_P_R
      fd9r        =     WQCBM3DA_fd9r(RegionNum)
      fg9r        =     WQCBM3DA_fg9r(RegionNum)

Return


Entry OpenWQCBM3DTVRCFiles
```

```
            NumTVRCFiles = WQCBM3D_NumTVRCFiles
            Allocate(WQCBM3D_nTVRCclms(NumTVRCFiles), Stat = AllocError)
            Allocate(WQCBM3D_iTVRCamp(NumTVRCFiles,256),
WQCBM3D_iTVRCcol(NumTVRCFiles,256), WQCBM3D_iTVRCunit(NumTVRCFiles,256),
Stat = AllocError)
            Allocate(WQCBM3D_TVRCValue(2,256), Stat = AllocError)
            Allocate(WQCBM3D_TVRCSTime(NumTVRCFiles), Stat = AllocError)
            Allocate(WQCBM3D_TVRCETime(NumTVRCFiles), Stat = AllocError)
            Allocate(WQCBM3D_TVRCInterp(NumTVRCFiles), Stat = AllocError)
            Allocate(WQCBM3D_TVRCMVNumber(NumTVRCFiles), Stat =
AllocError)
            Allocate(WQCBM3D_EndFileCountKDG(NumTVRCFiles), Stat =
AllocError)

            WQCBM3D_nTVRCclms = 0
            WQCBM3D_iTVRCamp = 0.0
            WQCBM3D_iTVRCcol = 0
            WQCBM3D_TVRCValue = 0.0
            WQCBM3D_TVRCInterp = 0
            WQCBM3D_TVRCMVNumber = 999999999.0
            WQCBM3D_EndFileCountKDG = 0

            Do jdd = 1, NumTVRCFiles
                  If(WQCBM3D_UseTVRCFile(jdd) == 0) Cycle
                  message = Trim(WQCBM3D_TVRCFileName(jdd))
                  If(message(1:len_trim(message)-1) == 'No_Data_File')
Cycle
                  file_name = message(1:len_trim(message)-1)
                  Call file_exist(file_name, ierr)
                  If (ierr < 0) Then
                        message = 'Error in
opening'//file_name(1:len_trim(file_name))//Char(0)
                        iretw =
MessageBox(GetActiveWindow(),message,'GLLVHT
Model'C,MB_ICONERROR.or.MB_OK)
                        Write(NFErr,'(a)') message(1:len_trim(message))
                        GLLVHTError = 1
                        Return
                  End If
                  Write(*,'(a)')
'Initializing'//file_name(1:len_trim(file_name))
                  Call GetUnitNumber(NFWQCBM3D(jdd))
                  Open (NFWQCBM3D(jdd), file = file_name, Status='old',
SHARED, Action = 'READ')
                  ReWind(NFWQCBM3D(jdd))
                  message = ''
                  Read (NFWQCBM3D(jdd),'(a)') message

                  VersionCheck = 0
                  If(Index(message,'V1') /= 0) VersionCheck = 1
                  If(Index(message,'V2') /= 0) VersionCheck = 2
                  If(Index(message,'V3') /= 0) VersionCheck = 3
                  If(Index(message,'V4') /= 0) VersionCheck = 4
```

```
                  Write(*,*) '.KDG TVRC File Number =', jdd, 'Version
Number = ', VersionCheck

                  Do While (.true.)
                        Read (NFWQCBM3D(jdd),'(a)') message
                        If(Index(message,'$') == 0) Exit
                  End Do
                  BackSpace(NFWQCBM3D(jdd))

                  Read(NFWQCBM3D(jdd),*) Value1, Value2
                  Read(NFWQCBM3D(jdd),*) BytVal1
                  Read(NFWQCBM3D(jdd),*) WQCBM3D_nTVRCclms(jdd)

                  Do i = 1, WQCBM3D_nTVRCclms(jdd)
                        Read(NFWQCBM3D(jdd),*) WQCBM3D_iTVRCCol(jdd,i),
BytVal1, BytVal2, WQCBM3D_iTVRCAmp(jdd,i), BytVal3, message
                        If(WQCBM3D_iTVRCCol(jdd,i) == SkipTVDColumnValue)
Cycle
                        WQCBM3D_iTVRCUnit(jdd,i) = BytVal3
                  End Do
                  !
                  !
                  Do While (.true.)
                        Read (NFWQCBM3D(jdd),'(a)') message
                        If(index(message,'$') == 0) Exit
                  End Do
                  Backspace(NFWQCBM3D(jdd))
                  Read(NFWQCBM3D(jdd),*,iostat = istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&

(WQCBM3D_TVRCValue(1,i), i = 1, WQCBM3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFWQCBM3D(jdd),WQCBM3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) WQCBM3D_EndFileCountKDG(jdd) =
WQCBM3D_EndFileCountKDG(jdd) + 1
                  WQCBM3D_TVRCSTime(jdd) = dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin)

                  Read(NFWQCBM3D(jdd),*,iostat = istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&

(WQCBM3D_TVRCvalue(2,i), i = 1, WQCBM3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFWQCBM3D(jdd),WQCBM3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) WQCBM3D_EndFileCountKDG(jdd) =
WQCBM3D_EndFileCountKDG(jdd) + 1
```

```
                  BackSpace(NFWQCBM3D(jdd))
                  WQCBM3D_TVRCETime(jdd) = dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin)

                  If (mdltime < WQCBM3D_TVRCSTime(jdd)) Then
                        Write(NFlog,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFlog,'(a)') 'File Name:
'//WQCBM3D_TVRCFileName(jdd)
                        Write(NFlog,'(a)') 'WQ MOdel : '//'WQCBM'
                        Write(NFlog,'(/)')

                        Write(NFcle,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFcle,'(a)') 'File Name:
'//WQCBM3D_TVRCFileName(jdd)
                        Write(NFcle,'(a)') 'WQ MOdel : '//'WQCBM'
                        Write(NFcle,'(/)')
                  End If
            End Do

      Return

      Entry ReadWQCBM3DTVRCFiles(jd)

            If(WQCBM3D_NumTVRCFiles == 0) Return
            If(.Not. WQCBM3D_UseTVRCFile(jd)) Return

            Do While (.true. .and. EndFileCountWDG(jd) <= 1)
                  !
                  !
                  If(MdlTime >= WQCBM3D_TVRCSTime(jd) .and. MdlTime <=
WQCBM3D_TVRCETime(jd)) Then

                        !SP 10/27/2007 added to re-read the values
                        Backspace(NFWQCBM3D(jd))
                        Read(NFWQCBM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (WQCBM3D_TVRCvalue(1,nci), nci = 1,
WQCBM3D_nTVRCclms(jd))

                        Read(NFWQCBM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (WQCBM3D_TVRCvalue(2,nci), nci = 1,
WQCBM3D_nTVRCclms(jd))
                        Backspace(NFWQCBM3D(jd))
                        !End SP 10/27/2007 added to re-read the values

                        Factor = 0.0
                        Select Case(WQCBM3D_TVRCInterp(jd))
                              Case(0)
                                    Factor = 0.0
                              Case(1)
```

```
                                    Factor = (MdlTime -
WQCBM3D_TVRCSTime(jd))/(WQCBM3D_TVRCETime(jd) - WQCBM3D_TVRCSTime(jd))
                              Case(2)
                                    Factor = 1.0
                        End Select
                        !
                        !Linear interpolation in both depth and time
                        Do nci = 1,WQCBM3D_NTVRCclms(jd)
                              Value1 = WQCBM3D_TVRCvalue(1,nci)
                              Value2 = WQCBM3D_TVRCvalue(2,nci)
                              If(Value1 /= WQCBM3D_TVRCMVNumber(jd) .and.
Value2 /= WQCBM3D_TVRCMVNumber(jd)) Then
                                    Value = Factor*Value2 + (1.0 -
Factor)*Value1
                              Else If(Value1 == WQCBM3D_TVRCMVNumber(jd)
.and. Value2 /= WQCBM3D_TVRCMVNumber(jd)) Then
                                    Value = Value2
                              Else If(Value1 /= WQCBM3D_TVRCMVNumber(jd)
.and. Value2 == WQCBM3D_TVRCMVNumber(jd)) Then
                                    Value = Value1
                              End If
                              If(WQCBM3D_iTVRCcol(jd,nci) ==
SkipTVDColumnValue) Cycle
                              Select Case(WQCBM3D_iTVRCcol(jd,nci))
                                    !
                                    !NH3
                                    Case(1)
                                          anc         =     Value
                                    Case(2)
                                          k71         =
      RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                                    Case(3)
                                          th71  =     Value
                                    Case(4)
                                          k12         =
      RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                                    Case(5)
                                          th12  =     Value
                                    Case(6)
                                          knit  =     Value
                                    Case(7)
                                          kmnc    =   Value
                                    !
                                    !NO3
                                    Case(8)
                                          k2d         =
      RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                                    Case(9)
                                          th2d  =     Value
                                    Case(10)
                                          kno3  =     Value
                                    !
                                    !PO4
                                    Case(11)
```

```
                              apc         =     Value
                        Case(12)
                              k83         =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(13)
                              th83  =     Value
                        Case(14)
                              kmpc  =     Value
                        Case(15)
                              plc         =     Value
                        !
                        !Diatoms
                        Case(16)
                              c2chla_d    =     Value
                        Case(17)
                              rins_d            =     Value
                        Case(18)
                              kmn_d       =     Value
                        Case(19)
                              ZPGMode_d   =   Value
                        Case(20)
                              kgmicro_d   =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(21)
                              kgmacro_d   =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(22)
                              thkt_d            =     Value
                        Case(23)
                              k1d_d       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(24)
                              k1c_d       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(25)
                              th1c_d            =     Value
                        Case(26)
                              kmp_d       =     Value

                        Case(27)
                              k1r_d       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(28)
                              th1r_d            =     Value
                        Case(29)
                              vs4_d       =
VelocityConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(30)
                              fe_d        =     Value
                        Case(31)
                              as_d        =     Value
                        !
                        !Dynoflagellates
                        Case(32)
```

```
                              c2chla_f    =     Value
                           Case(33)
                              rins_f           =     Value
                           Case(34)
                              kmn_f       =     Value
                           Case(35)
                              ZPGMode_f   =     Value
                           Case(36)
                              kgmicro_f   =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(37)
                              kgmacro_f   =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(38)
                              thkt_f           =     Value
                           Case(39)
                              k1d_f       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(40)
                              k1c_f       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(41)
                              th1c_f           =     Value
                           Case(42)
                              kmp_f            =     Value
                           Case(43)
                              k1r_f       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(44)
                              th1r_f           =     Value
                           Case(45)
                              vs4_f       =
VelocityConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                           Case(46)
                              fe_f        =     Value
                           Case(47)
                              as_f        =     Value
                           !
                           !DO
                           Case(48)
                              SDOERMethod =   Value
                           Case(49)
                              kdf              =     Value
                           Case(50)
                              kds              =     Value
                           Case(51)
                              ReAerationFactor = Value
                           !
                           !CBOD_F
                           Case(52)
                              aoc              =     Value
                           Case(53)
                              thd              =     Value
                           Case(54)
```

```
                              kbod        =     Value
                        Case(55)
                              foc         =     Value
                        !
                        !CBOD_S
                        Case(56)
                              fd5         =     Value
                        !
                        !ON_D
                        Case(57)
                              kh7p        =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(58)
                              thh7p       =     Value
                        Case(59)
                              fon         =     Value
                        !
                        !ON_P
                        Case(60)
                              vs7         =
VelocityConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(61)
                              ancp        =     Value
                        !
                        !OP_D
                        Case(62)
                              kh8p        =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(63)
                              thh8p       =     Value
                        Case(64)
                              fop         =     Value
                        !
                        !OP_P
                        Case(65)
                              vs8         =
VelocityConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(66)
                              apcp        =     Value
                        !
                        !OC_P_F


                        Case(67)
                              fd9f        =     Value
                        Case(68)
                              fg9f        =     Value
                        Case(69)
                              kpd9f       =
RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                        Case(70)
                              thpd9f      =     Value
                        Case(71)
```

```
                                    vs9               =
      VelocityConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                                  !
                                  !OC_P_S
                                  Case(72)
                                        fd9s        =     Value
                                  Case(73)
                                        fg9s        =     Value
                                  Case(74)
                                        kpd9s       =
      RateConversion(Value, WQCBM3D_iTVRCUnit(jd,nci))
                                  Case(75)
                                        thpd9s          =   Value
                                  !
                                  !OC_P_R
                                  Case(76)
                                        fd9r        =     Value
                                  Case(77)
                                        fg9r        =     Value
                            End Select
                      End Do
                      Exit
                Else
                      !
                      !vsk 01/04/00
                      If(WQCBM3D_TVRCSTime(jd) > MdlTime) Return
                      Read(NFWQCBM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
               (WQCBM3D_TVRCvalue(1,nci), nci = 1,
WQCBM3D_nTVRCclms(jd))

                      Call
seterrormessage(istat,NFWQCBM3D(jd),WQCBM3D_TVRCFileName(jd))
                      If(istat < 0) WQCBM3D_EndFileCountKDG(jd) =
WQCBM3D_EndFileCountKDG(jd) + 1
                      If(istat > 0) then
                            GLLVHTError = 1
                            Return
                      End If

                      WQCBM3D_TVRCSTime(jd) =
dfloat(dttm2julian(tvdsyear, tvdsmonth, tvdsday, tvdshour, tvdsmin))

                      Read(NFWQCBM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
               (WQCBM3D_TVRCvalue(2,nci), nci = 1,
WQCBM3D_nTVRCclms(jd))
                      Backspace(NFWQCBM3D(jd))
                      Call
seterrormessage(istat,NFWQCBM3D(jd),WQCBM3D_TVRCFileName(jd))
                      If(istat < 0) WQCBM3D_EndFileCountKDG(jd) =
WQCBM3D_EndFileCountKDG(jd) + 1
                      If(istat > 0) Then
                            GLLVHTError = 1
```

```
                              Return
                          End If
                          WQCBM3D_TVRCETime(jd) =
dfloat(dttm2julian(tvdsyear, tvdsmonth, tvdsday, tvdshour, tvdsmin))
                    End If
               End Do  !jd

      Return

Return
End SUBROUTINE WQCBM3DSSTerms
```

GAM Module

```
Subroutine GAMSubs
      !
      !Global variables
      Use dfwin
      Use G7AllocMemory
      Use GEMSS_GAMAllocMemoryVariables
      Use AllocMemoryWQCBM3DVariables
      Use AllocMemoryWQADD3DVariables
      Include 'G7Variables.f90'
      Include 'GEMSS_GAMVariables.f90'
      Include 'G7WQADD3DVariables.f90'

      Character(256) message1, message2
      Real(8) Depth, PhytoConc, Depth1
      Real(8) d1, Sum
      Real(8) df1, ftmp, po4avl

      Real(8) MAlgGrowth, MAlgResp, MAlgDeath, MAlgExrec, MAlgGraze,
MAlgSettl
      Real(8) NH3FromAlg, NH3ToAlg, NO3ToAlg
      Real(8) PO4FromAlg, PO4ToAlg
      Real(8) DOFromAlg, DOToAlg
      Real(8) FCBODFromAlg, SCBODFromAlg, POCFFromAlg, POCSFromAlg,
POCRFromAlg
      Real(8) DONFromAlg, DOPFromAlg, PONFromAlg, POPFromAlg
      Real(8) GAM3D_ke, stox, Temp20
      Real(8) AlgaeGrwthNetLimit

      Real(8) apc, anc, ancp, apcp, aoc
      Real(8) fd9f, fg9f, fd9s, fg9s, fd9r, fg9r
      Integer(2) i, j, nc, k, IntVal1, id, TVRCFileCount, jj, kn
      Integer(4) n
      Real(8) Num, Den, Alpha0, Alpha1

      Logical(1) ChangesInRegions
      Real(8) Pam

      Byte  Units
      Real(8) RateConversion, LightConversion, VelocityConversion

      Integer(2)  NumTVRCFiles, jdd, jd, nci, nj,ni
      Integer(4)  ierr, iretw, DTTM2JULIAN, IStat
      Byte        VersionCheck, BytVal1, BytVal2, BytVal3,d11
      Real(8)           Value1, Value2, Factor, Value
      Byte        GAM3D_UseTVRCFileLocal
      Character(256) GAM3D_TVRCFileNameLocal

      Byte  InsideGAM
      !
      !Local variables for Alkalinity, pH and Conductivity
      Real(8) ralkaa, ralkan, ralkbn, ralkbp, ralkn, ralkden, rondn
      Real(8) rcca, rcco, rccd, rccc, ralkda, ralkdn, defa
      Real(8) ph_k1, ph_k2, ph_kw, ph_kh
      Real(8) alp0, alp1
```

```
      Real(8)     kacT, ron, roa, aca, ph_hh
      Real(8) CO2Sat  ! CO2, HCO3, CO3, TIOC, ALKL
      Real(8) prefam

      Common/GAMLocalVariables/&
                  Depth, PhytoConc, Depth1,&
                  Sum,&
                  df1, ftmp, po4avl,&
                  MAlgGrowth, MAlgResp, MAlgDeath, MAlgExrec, MAlgGraze,
MAlgSettl,&
                  NH3FromAlg, NH3ToAlg, NO3ToAlg,&
                  PO4FromAlg, PO4ToAlg,&
                  DOFromAlg, DOToAlg,&
                  FCBODFromAlg, SCBODFromAlg, POCFFromAlg, POCSFromAlg,
POCRFromAlg,&
                  DONFromAlg, DOPFromAlg, PONFromAlg, POPFromAlg,&
                  GAM3D_ke, stox, Temp20, id,&
                  NumTVRCFiles, jdd, IStat, nci, nj,ni,&
                  ierr, iretw,&
                  VersionCheck, BytVal1, BytVal2,BytVal3,&
                  Value1, Value2, Factor, Value,d11

      Common/CommonVariablesBetweenGAMandWQCBM/&
                  apc, anc, ancp, apcp, aoc,&
                  fd9f, fg9f, fd9s, fg9s, fd9r, fg9r, InsideGAM

      !
      !
      !This line should be there in all water quality models
      Common/CommonVariablesForAllWQModels/&
                  i, j, k, n, nc, d1

      Common/CommonVariablesBetweenGAMandWQADD3D/&
                  Pam

      !
      !pH
      Common/CommonVariablesForpH/&
                  ralkaa, ralkan, ralkbn, ralkbp, ralkn, ralkden, rondn,&
                  rcca, rcco, rccd, rccc, ralkda, ralkdn, defa,&
                  ph_k1, ph_k2, ph_kw, ph_kh,&
                  alp0, alp1,&
                  kacT, ron, roa, aca, ph_hh,&
                  CO2Sat,&
                  prefam
      Data SetTop /0/


      Save

      Return

      Entry ReadGAMControlFile
```

```
            If(.Not. GenAlgaeModel) Return

            TVRCFileCount = 0
            j = 1
         Do While(.true.)
             read (NFctl,*,end=1000) Message
             Select Case(message(1:len_trim(message)-1))
                       !
                       !Look for iGAM
                       Case('iGAM')
                    BackSpace(NFctl)
                    Read(NFctl,*) message1, message2, IntVal1, IntVal1,
IntVal1, NUMGAM3DRegions
                              Call AllocateMemoryForGAMVariables
                              Do i = 1, ngamcs
                                    Read(NFctl,'(2(/))')
                                    Read(NFctl,*) message1, message2,
GAM3D_UseNutrientLimit(j)
                                    Read(NFctl,*) message1, message2,
GAM3D_UseTempLimit(j)
                                    Read(NFctl,*) message1, message2,
GAM3D_UseSalineToxicLimit(j)
                                    Read(NFctl,*) message1, message2,
GAM3D_UseLightLimit(j)
                                    Read(NFctl,*) message1, message2,
GAM3D_k1r(i,j), Units
                                    GAM3D_k1r(i,j) =
RateConversion(GAM3D_k1r(i,j), Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_Tht_k1r(i,j)

                                    Read(NFctl,*) message1, message2,
GAM3D_k1c(i,j), Units
                                    GAM3D_k1c(i,j) =
RateConversion(GAM3D_k1c(i,j), Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_Tht_k1c(i,j)

                                    Read(NFctl,*) message1, message2,
GAM3D_k1d(i,j)
                                    GAM3D_k1d(i,j) =
RateConversion(GAM3D_k1d(i,j), Units)

                                    Read(NFctl,*) message1, message2,
GAM3D_fe(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_as(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_ws(i,j), Units
                                    GAM3D_ws(i,j) =
VelocityConversion(GAM3D_ws(i,j), Units)

                                    Read(NFctl,*) message1, message2,
GAM3D_ZPGMode(i,j)
```

```
                                    Read(NFctl,*) message1, message2,
GAM3D_kgmicro(i,j), Units
                                    GAM3D_kgmicro(i,j) =
RateConversion(GAM3D_kgmicro(i,j), Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_Tht_kgmicro(i,j)

                                    Read(NFctl,*) message1, message2,
GAM3D_kgmacro(i,j), Units
                                    GAM3D_kgmacro(i,j) =
RateConversion(GAM3D_kgmacro(i,j), Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_Tht_kgmacro(i,j)

                                    Read(NFctl,*) message1, message2,
GAM3D_cchl(i,j)

                                    Read(NFctl,*) message1, message2,
GAM3D_LightModel(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_kke(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_kechl(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_ISat(i,j), Units
                                    GAM3D_ISat(i,j) =
LightConversion(GAM3D_ISat(i,j),Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_khn(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_khp(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_stMethod(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_stf(i,j), Units
                                    !
                                    !Ambrose Bug Fix:
                                    GAM3D_stf(i,j) =
RateConversion(GAM3D_stf(i,j), Units)
                                    Read(NFctl,*) message1, message2,
GAM3D_khst(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_tm(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_ktg1(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_ktg2(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_fd5(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_fon(i,j)
                                    Read(NFctl,*) message1, message2,
GAM3D_fop(i,j)
```

```
                                    Read(NFctl,*) message1, message2,
GAM3D_foc(i,j)
                              End Do
                              !
                              !VSK 05-02-2005;VSK 06-07-2010
                              Read(NFctl,'(//)')
                              Read(NFctl,*) message1, message2,
GAM3D_UseTVRCFileLocal        !GAM3D_UseTVRCFile(j)
                              Read(NFctl,*) message1, message2,
GAM3D_TVRCFileNameLocal         !GAM3D_TVRCFileName(j)
                              Exit
                    End Select
               End Do
               !
               !Use region rates and constants
               Read(NFctl,*) message1, NUMGAM3DRegions

               TVRCFileCount = TVRCFileCount + 1
               GAM3D_NumTVRCFiles = NumGAM3DRegions + 1

               If(GAM3D_NumTVRCFiles /= 0) Then
                    Allocate(GAM3D_UseTVRCFile(GAM3D_NumTVRCFiles), Stat =
AllocError)
                    Allocate(GAM3D_TVRCFileName(GAM3D_NumTVRCFiles), Stat =
AllocError)
                    Allocate(NFGAM3D(GAM3D_NumTVRCFiles), Stat = AllocError)
                    Allocate(GAM3DRegionName(NumGAM3DRegions), Stat =
AllocError)
               End If

               GAM3D_TVRCFileName(1) = GAM3D_TVRCFileNameLocal
               GAM3D_UseTVRCFile(1)  = GAM3D_UseTVRCFileLocal

               If(NUMGAM3DRegions > 0) Then
                    Allocate (GAM3DIStart(NUMGAM3DRegions),
GAM3DJStart(NUMGAM3DRegions), Stat = AllocError)
                    Allocate(GAM3DIEnd(NUMGAM3DRegions),
GAM3DJEnd(NUMGAM3DRegions), Stat = AllocError)
                    Allocate(GAM3DRegionStatus(im+1,jm+1), Stat =
AllocError)
                    Allocate(GAM3DRegionName(NUMGAM3DRegions), Stat =
AllocError)
                    Allocate(GAM3DRegionRCDStatus(NumGAM3DRegions), Stat =
AllocError)
                    GAM3DRegionRCDStatus = 0
                    GAM3DRegionStatus = 0
                    GAM3DIStart = 0
                    GAM3DJStart = 0
                    GAM3DIEnd = 0
                    GAM3DJEnd = 0
               End If

               ChangesInRegions = .False.
               Do k = 1, NUMGAM3DRegions
```

```
                  Read(NFctl,*) GAM3DRegionName(k), GAM3DIStart(k),
GAM3DIEnd(k), GAM3DJStart(k), GAM3DJEnd(k), i, j, GAM3DRegionRCDStatus(k)
                  Do i = GAM3DIStart(k)+1,GAM3DIEnd(k)+1
                        Do j = GAM3DJStart(k)+1,GAM3DJEnd(k)+1
                              GAM3DRegionStatus(i,j) = k
                        End Do
                  End Do
                  If(GAM3DRegionRCDStatus(k) == 1) Then
                        ChangesInRegions = .True.
                  End If
            End Do

            If(.Not. ChangesInRegions) NUMGAM3DRegions = 0

            Do jj = 1, NUMGAM3DRegions
                  Read(NFctl,*) message1, message, message
                  j = jj + 1
                  Do i = 1, ngamcs
                        Read(NFctl,'(2(/))')
                        !
                        !Skip switches as there is no need to read them
for different regions
                        Read(NFctl,*) message1, message2,
GAM3D_UseNutrientLimit(j)
                        Read(NFctl,*) message1, message2,
GAM3D_UseTempLimit(j)
                        Read(NFctl,*) message1, message2,
GAM3D_UseSalineToxicLimit(j)
                        Read(NFctl,*) message1, message2,
GAM3D_UseLightLimit(j)

                        Read(NFctl,*) message1, message2, GAM3D_k1r(i,j)
                        GAM3D_k1r(i,j) = RateConversion(GAM3D_k1r(i,j),
Units)
                        Read(NFctl,*) message1, message2,
GAM3D_Tht_k1r(i,j)

                        Read(NFctl,*) message1, message2, GAM3D_k1c(i,j),
Units
                        GAM3D_k1c(i,j) = RateConversion(GAM3D_k1c(i,j),
Units)
                        Read(NFctl,*) message1, message2,
GAM3D_Tht_k1c(i,j)

                        Read(NFctl,*) message1, message2, GAM3D_k1d(i,j)
                        GAM3D_k1d(i,j) = RateConversion(GAM3D_k1d(i,j),
Units)

                        Read(NFctl,*) message1, message2, GAM3D_fe(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_as(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_ws(i,j),
Units
                        GAM3D_ws(i,j) = VelocityConversion(GAM3D_ws(i,j),
Units)
```

```
                        Read(NFctl,*) message1, message2,
GAM3D_ZPGMode(i,j)
                        Read(NFctl,*) message1, message2,
GAM3D_kgmicro(i,j), Units
                        GAM3D_kgmicro(i,j) =
RateConversion(GAM3D_kgmicro(i,j), Units)
                        Read(NFctl,*) message1, message2,
GAM3D_Tht_kgmicro(i,j)

                        Read(NFctl,*) message1, message2,
GAM3D_kgmacro(i,j), Units
                        GAM3D_kgmacro(i,j) =
RateConversion(GAM3D_kgmacro(i,j), Units)
                        Read(NFctl,*) message1, message2,
GAM3D_Tht_kgmacro(i,j)

                        Read(NFctl,*) message1, message2, GAM3D_cchl(i,j)

                        Read(NFctl,*) message1, message2,
GAM3D_LightModel(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_kke(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_kechl(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_ISat(i,j),
Units
                        GAM3D_ISat(i,j) =
LightConversion(GAM3D_ISat(i,j),Units)
                        Read(NFctl,*) message1, message2, GAM3D_khn(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_khp(i,j)
                        Read(NFctl,*) message1, message2,
GAM3D_stMethod(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_stf(i,j),
Units
                        GAM3D_stf(i,j) = RateConversion(GAM3D_stf(i,j),
Units)
                        Read(NFctl,*) message1, message2, GAM3D_khst(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_tm(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_ktg1(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_ktg2(i,j)

                        Read(NFctl,*) message1, message2, GAM3D_fd5(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_fon(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_fop(i,j)
                        Read(NFctl,*) message1, message2, GAM3D_foc(i,j)
                End Do
                !
                !VSK 05-02-2005; VSK 06-07-2010
                Read(NFwqctl,'(//)')
                TVRCFileCount = TVRCFileCount + 1
                Read(NFctl,*) message1, message2,
GAM3D_UseTVRCFile(TVRCFileCount)
                Read(NFctl,*) message1, message2,
GAM3D_TVRCFileName(TVRCFileCount)
            End Do
```

```
            !
            !
            If(GAM3D_NumTVRCFiles /= 0) Call OpenGAM3DTVRCFiles

            Return
            1000 message = 'Error in Reading Rates and Constants for
WQDPM, Please check the control file'C
            iretw = MessageBox(GetActiveWindow(),message,'GEMSS
Model'C,MB_ICONINFORMATION.or.MB_OK)
            Stop
      Return


      Entry AllocateMemoryForGAMVariables
            !
            !Rates and constants variables
            Allocate (GAM3D_k1r(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_k1r = 0.0d+00
            Allocate (GAM3D_Tht_k1r(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_Tht_k1r = 0.0d+00
            Allocate (GAM3D_k1c(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_k1c = 0.0d+00
            Allocate (GAM3D_Tht_k1c(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_Tht_k1c = 0.0d+00
            Allocate (GAM3D_k1d(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_k1d = 0.0d+00
            Allocate (GAM3D_Tht_k1d(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_Tht_k1d = 0.0d+00
            Allocate (GAM3D_fe(ngamcs,NUMGAM3DRegions+1),    Stat =
AllocError)
            GAM3D_fe = 0.0d+00
            Allocate (GAM3D_as(ngamcs,NUMGAM3DRegions+1),    Stat =
AllocError)
            GAM3D_as = 0.0d+00
            Allocate (GAM3D_ZPGMode(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_ZPGMode = 0
            Allocate (GAM3D_kgmicro(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_kgmicro = 0.0d+00
            Allocate (GAM3D_Tht_kgmicro(ngamcs, NUMGAM3DRegions+1), Stat
= AllocError)
            GAM3D_Tht_kgmicro = 0.0d+00
            Allocate (GAM3D_kgmacro(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_kgmacro = 0.0d+00
```

```
            Allocate (GAM3D_Tht_kgmacro(ngamcs, NUMGAM3DRegions+1), Stat
= AllocError)
            GAM3D_Tht_kgmacro = 0.0d+00
            Allocate (GAM3D_fd5(ngamcs,NUMGAM3DRegions+1),   Stat =
AllocError)
            GAM3D_fd5 = 0.0d+00
            Allocate (GAM3D_fon(ngamcs,NUMGAM3DRegions+1),   Stat =
AllocError)
            GAM3D_fon = 0.0d+00
            Allocate (GAM3D_fop(ngamcs,NUMGAM3DRegions+1),   Stat =
AllocError)
            GAM3D_fop = 0.0d+00
            Allocate (GAM3D_foc(ngamcs,NUMGAM3DRegions+1),   Stat =
AllocError)
            GAM3D_foc = 0.0d+00
            Allocate (GAM3D_ws(ngamcs,NUMGAM3DRegions+1),    Stat =
AllocError)
            GAM3D_ws = 0.0d+00
            Allocate (GAM3D_kechl(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_kechl = 0.0d+00
            Allocate (GAM3D_cchl(ngamcs,NUMGAM3DRegions+1),  Stat =
AllocError)
            GAM3D_cchl = 0.0d+00
            Allocate (GAM3D_LightModel(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_LightModel = 0
            Allocate (GAM3D_ISat(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_ISat = 0.0d+00
            Allocate (GAM3D_khn(ngamcs,NUMGAM3DRegions+1),   Stat =
AllocError)
            GAM3D_khn = 0.0d+00
            Allocate (GAM3D_khp(ngamcs,NUMGAM3DRegions+1),  Stat =
AllocError)
            GAM3D_khp = 0.0d+00
            Allocate (GAM3D_stf(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_stf = 0.0d+00
            Allocate (GAM3D_khst(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_khst = 0.0d+00
            Allocate (GAM3D_stMethod(ngamcs, NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_stMethod = 0
            Allocate (GAM3D_tm(ngamcs,NUMGAM3DRegions+1),    Stat =
AllocError)
            GAM3D_tm = 0.0d+00
            Allocate (GAM3D_ktg1(ngamcs,NUMGAM3DRegions+1),  Stat =
AllocError)
            GAM3D_ktg1 = 0.0d+00
            Allocate (GAM3D_ktg2(ngamcs,NUMGAM3DRegions+1),  Stat =
AllocError)
            GAM3D_ktg2 = 0.0d+00
```

```
            Allocate (GAM3D_kke(ngamcs,NUMGAM3DRegions+1), Stat =
AllocError)
            GAM3D_kke = 0.0d+00
            !
            !Computed variables
            Allocate (GAM3D_dr(ngamcs), Stat = AllocError)
            GAM3D_dr = 0.0d+00
            Allocate (GAM3D_gp(ngamcs), Stat = AllocError)
            GAM3D_gp = 0.0d+00
            Allocate (GAM3D_dd(ngamcs), Stat = AllocError)
            GAM3D_dd = 0.0d+00
            Allocate (GAM3D_pnh3(ngamcs), Stat = AllocError)
            GAM3D_pnh3 = 0.0d+00
            Allocate (GAM3D_rcchl(ngamcs), Stat = AllocError)
            GAM3D_rcchl = 0.0d+00
            Allocate (GAM3D_gp(ngamcs), Stat = AllocError)
            GAM3D_gp = 0.0d+00
            Allocate (GAM3D_kess(ngamcs), Stat = AllocError)
            GAM3D_kess = 0.0d+00
            Allocate (GAM3D_isc(ngamcs), Stat = AllocError)
            GAM3D_isc = 0.0d+00
            Allocate (GAM3D_nl(ngamcs), Stat = AllocError)
            GAM3D_nl = 0.0d+00
            Allocate (GAM3D_pl(ngamcs), Stat = AllocError)
            GAM3D_pl = 0.0d+00
            Allocate (GAM3D_apc(ngamcs), Stat = AllocError)
            GAM3D_apc = 0.0d+00
            Allocate (GAM3D_fn(ngamcs), Stat = AllocError)
            GAM3D_fn = 0.0d+00
            Allocate (GAM3D_ft(ngamcs), Stat = AllocError)
            GAM3D_ft = 0.0d+00
            Allocate (GAM3D_fi(ngamcs), Stat = AllocError)
            GAM3D_fi = 0.0d+00
            Allocate (GAM3D_ftox(ngamcs), Stat = AllocError)
            GAM3D_ftox = 0.0d+00
            Allocate (GAM3D_pnh3(ngamcs), Stat = AllocError)
            GAM3D_pnh3 = 0.0d+00
            Allocate (GAM3D_SumPhyt(n1+1), Stat = AllocError)
            GAM3D_SumPhyt = 0.0d+00
            Allocate (GAM3D_ri(km_p+1), Stat = AllocError)
            GAM3D_ri = 0.0d+00
            Allocate (GAM3D_ggraze(ngamcs), Stat = AllocError)
            GAM3D_ggraze = 0.0d+00
            Allocate (GAM3D_gmicro(ngamcs), Stat = AllocError)
            GAM3D_gmicro = 0.0d+00
            Allocate (GAM3D_gmacro(ngamcs), Stat = AllocError)
            GAM3D_gmacro = 0.0d+00

            Allocate(GAM3D_UseNutrientLimit(NUMGAM3DRegions+1), Stat =
AllocError)
            Allocate(GAM3D_UseTempLimit(NUMGAM3DRegions+1), Stat =
AllocError)
            Allocate(GAM3D_UseSalineToxicLimit(NUMGAM3DRegions+1), Stat =
AllocError)
```

```
            Allocate(GAM3D_UseLightLimit(NUMGAM3DRegions+1), Stat =
AllocError)

            Allocate(GAM3D_UDC_LimitingFactor(8,5))

            Allocate(GAM3D_UDC_Output(2,13))
            GAM3D_UDC_LimitingFactor = 0.d00
            GAM3D_UDC_Output = 0.d00

            Allocate(GAM_3D_fn(n1+1,km_p+1,ngamcs),
GAM_3D_fi(n1+1,km_p+1,ngamcs), GAM_3D_ft(n1+1,km_p+1,ngamcs))
            Allocate(GAM_3D_ftox(n1+1,km_p+1,ngamcs),
GAM_3D_fnet(n1+1,km_p+1,ngamcs))
            GAM_3D_fn = 0.d+00
            GAM_3D_fi = 0.d+00
            GAM_3D_ft = 0.d+00
            GAM_3D_ftox = 0.d+00
            GAM_3D_fnet = 0.d+00
            Allocate(GAM_3D_N_lim(n1+1,km_p+1,ngamcs), Stat = AllocError)
            Allocate(GAM_3D_P_lim(n1+1,km_p+1,ngamcs), Stat = AllocError)
            Allocate(GAM_3D_T_lim(n1+1,km_p+1,ngamcs), Stat = AllocError)
            GAM_3D_N_lim = 0.0d+00
            GAM_3D_P_lim = 0.0d+00
            GAM_3D_T_lim = 0.0d+00
            Allocate(GAM_3D_PPROD(n1+1,km_p+1,ngamcs), Stat = AllocError)
            GAM_3D_PPROD = 0.0

      Return


      Entry ComputeDepthRelatedSolarRadiation

            RegionNum = 1
            If(NUMGAM3DRegions > 0) RegionNum = GAM3DRegionStatus(i,j)  +
1
            Depth = 0.0
            kt = ktwb(cwbn(i,j))
            d11 = dzk(kt) - z(n)
            GAM3D_ri = 0.0
            Do kn = kt, k0(i,j)
                  PhytoConc = 0.0
                  Do id = 1, ngamcs
                        PhytoConc = PhytoConc + c(n,kn,I_GAM(id))
                  End Do
                  Depth  = Depth + d11
                  Depth1 = Depth - d11/2.0
                  GAM3D_SumPhyt(n) = GAM3D_SumPhyt(n) + PhytoConc*d11
                  Do id = 1, ngamcs
                        GAM3D_ke  = GAM3D_kke(id,RegionNum) +
GAM3D_kechl(id,RegionNum)*GAM3D_SumPhyt(n)/GAM3D_cchl(id,RegionNum)
                  End Do
                  GAM3D_ri(kn)  = SolRad*exp(-1.0*GAM3D_ke*Depth1)
                  d11 = dzk(kn+1)
            End Do !kn
```

```
      Return


      Entry SetGAMSSTermsOutsideWQM


            Do n = 1, n1
                  !
                  !Skipping all nonactive and boundary padded cells (used
for open boundary condition only)
                  If(nm(n) /= 1) Cycle
                  i = i1(n)
                  j = j1(n)
                  RegionNum = 1
                  If(NUMGAM3DRegions > 0) RegionNum =
GAM3DRegionStatus(i,j) + 1
                  !
                  !VSK: 06-24-2010 - To handle values coming from met
spatially varying boundary condition
                  If(MaxNumMetBcs > 0) SolRad =
MetBCrs(dsgdr(CellMetBCNum(n)))
                  kt = ktwb(cwbn(i,j))
                  d11 = dzk(kt) - z(n)
                  SetTop = 0.0d+00
                  Depth = 0.0
                  CellArea = dxx(i,j)*dyy(i,j)
                  Call ComputeDepthRelatedSolarRadiation
                  Do k = kt, k0(i,j)
                        !
                        !Wall boundary
                        If(m(n,k) == 0) Cycle
                        CellVolume = dxx(i,j)*dyy(i,j)*d11
                        Call GAMSSTermsInsideWQM
                        SetTop = 1.0
                        d11 = dzk(k+1)
                  End Do
            End Do

      Return

      Entry GAMSSTermsInsideWQM
            !
            !Switch to control SumPhyt to be initialized once
            If(InsideGAM == 0) GAM3D_SumPhyt = 0.0d+00
            InsideGAM = 1
            !
            !
            RegionNum = 1
            Temp20 = c(n,k,I_Temp) - 20.0
            If(NUMGAM3DRegions > 0) RegionNum = GAM3DRegionStatus(i,j) +
1
            Call ReadGAM3DTVRCFiles(RegionNum)
            If(UseLOTTMethod == 1) Call ComputeDepthRelatedSolarRadiation
```

```
            !
            !Light limitation
            GAM3D_fi = 1.0
            If(GAM3D_UseLightLimit(RegionNum) == 1) Then
                  Do id = 1, ngamcs
                        Select Case(GAM3D_LightModel(id,RegionNum))
                              !
                              !Half Saturation
                              Case(1)
                                    If(GAM3D_ISat(id,RegionNum) == 0) Then
                                          GAM3D_fi(id) = 1.0
                                    Else
                                          If(UseLOTTMethod == 1) Then
                                                GAM3D_fi(id) =
GAM3D_ri(k)/(GAM3D_ISat(id,RegionNum) + GAM3D_ri(k))
                                          Else
                                                !QUAL2Kw depth-integrated
form of half sat eqn in this layer
                                             !phil = 1 / (ke * depth(i)) *
Log((Isat_rch(i) + Iat(i)) / (Isat_rch(i) + Iat(i) * Exp(-ke *
depth(i))))
                                       GAM3D_fi(id) = 1.0 / KeCalc(n,k) *
Log((GAM3D_ISat(id,RegionNum)  + &
                                       RTopPAR(n,k)) / (GAM3D_ISat(id,RegionNum)
+ RTopPAR(n,k) * RExtPAR(n,k)))
                                        End if
                                       End If
                              !
                              !Smith's function
                              Case(2)
                                    If(GAM3D_ISat(id,RegionNum) == 0) Then
                                          GAM3D_fi(id) = 1.0
                                    Else
                                          If(UseLOTTMethod == 1) Then
                                                GAM3D_fi(id) =
GAM3D_ri(k)/sqrt(GAM3D_ISat(id,RegionNum)**2.0 + GAM3D_ri(k)**2.0)
                                          Else
                                                !QUAL2Kw depth-integrated
form of Smith eqn in this layer
                                                !num = Iat(i) / Isat_rch(i)
+ Sqr(1 + (Iat(i) / Isat_rch(i)) ^ 2)
                                                !den = Iat(i) * Exp(-ke *
depth(i)) / Isat_rch(i) + Sqr(1 + (Iat(i) * Exp(-ke * depth(i)) /
Isat_rch(i)) ^ 2)
                                                !phil = 1 / (ke * depth(i))
* Log(num / den)
                                                num = RTopPAR(n,k) /
GAM3D_ISat(id,RegionNum) + Sqrt(1.0 + (RTopPAR(n,k) /
GAM3D_ISat(id,RegionNum)) ** 2.0)
                                                den = RTopPAR(n,k) *
RExtPAR(n,k) / GAM3D_ISat(id,RegionNum) + &
                                                      Sqrt(1.0 +
(RTopPAR(n,k) * RExtPAR(n,k) / GAM3D_ISat(id,RegionNum)) ** 2.0)
```

```
                                                GAM3D_fi(id) = 1.0 /
KeCalc(n,k) * Log(num / den)
                              End if
                                    End If
                               !
                               !Steel's Equation
                               Case(3)
                                    If(GAM3D_ISat(id,RegionNum) == 0) Then
                                          GAM3D_fi(id) = 1.0
                                    Else
                                          If(UseLOTTMethod == 1) Then
                                                GAM3D_fi(id) =
(GAM3D_ri(k)/GAM3D_ISat(id,RegionNum))*exp(1.0 -
GAM3D_ri(k)/GAM3D_ISat(id,RegionNum))
                                          Else
                                       !QUAL2Kw depth-integrated form of
Steele eqn in this layer
                                         !alpha0 = Iat(i) / Isat_rch(i)
                                            !alpha1 = Iat(i) / Isat_rch(i)
* Exp(-ke * depth(i))
                                            !phil = Exp(1) * (Exp(-alpha1)
- Exp(-alpha0)) / (ke * depth(i))
                                            alpha0 = RTopPAR(n,k) /
GAM3D_ISat(id,RegionNum)
                                            alpha1 = RTopPAR(n,k) /
GAM3D_ISat(id,RegionNum) * RExtPAR(n,k)
                                            GAM3D_fi(id) = Exp(1.0) *
(Exp(-alpha1) - Exp(-alpha0)) / KeCalc(n,k)
                              End If
                                    End If
                         End Select
                   End Do
             End If
             !
             !Nutrient limitations
            !nitrogen availability for algal growth
             ftmp = c(n,k,I_NH3) + c(n,k,I_NO3)
             Do id = 1, ngamcs
                   If(GAM3D_khn(id,RegionNum) == 0)Then
                         GAM3D_nl(id) = 1.0
                   Else
                         GAM3D_nl(id) = ftmp/(GAM3D_khn(id,RegionNum) +
ftmp)
                   End If
             End Do
             !
             !Phosphorus availability for algal growth
             Do id = 1, ngamcs
                   If(GAM3D_khp(id,RegionNum) == 0)Then
                         GAM3D_pl(id) = 1.0
                   Else
                         GAM3D_pl(id) =
c(n,k,I_PO4)/(GAM3D_khp(id,RegionNum) + c(n,k,I_PO4))
                   End If
```

```
            End Do
            !
            !
            GAM3D_fn = 1.0
            If(GAM3D_UseNutrientLimit(RegionNum) == 1) Then
                  Do id = 1, ngamcs
                        GAM3D_fn(id) = min(GAM3D_nl(id),GAM3D_pl(id))
                  End Do
            End If
            !
            !Temperature effects
            GAM3D_ft = 1.0
            If(GAM3D_UseTempLimit(RegionNum) == 1) Then
                  Do id = 1, ngamcs
                        If (GAM3D_Tht_k1c(id,RegionNum) .gt. 1.0) Then
                              GAM3D_ft(id) =
GAM3D_Tht_k1c(id,RegionNum)**Temp20
                        Else
                              If (c(n,k,I_Temp) < GAM3D_tm(id,RegionNum))
Then
                                    GAM3D_ft(id) = exp(-
GAM3D_ktg1(id,RegionNum)*(c(n,k,I_Temp) - GAM3D_tm(id,RegionNum))**2)
                              Else
                                    GAM3D_ft(id) = exp(-
GAM3D_ktg2(id,RegionNum)*(GAM3D_tm(id,RegionNum) - c(n,k,I_Temp))**2)
                              End If
                        End If
                  End Do
            End If
            !
            !Saline effects
            GAM3D_ftox = 1.0
            If(GAM3D_UseSalineToxicLimit(RegionNum) == 1) Then
                  Do id = 1,ngamcs
                        stox = dmax1(0.0d+00,c(n,k,I_Saln))
                        Select Case(GAM3D_stMethod(id,RegionNum))
                              !
                              !non-dimensional factor
                              Case(1)
                                    GAM3D_ftox(id) =
GAM3D_khst(id,RegionNum)**2.0/(GAM3D_khst(id,RegionNum)**2.0 + stox**2.0)
                              !
                              !dimensional 1/sec
                              Case(2)
                                    GAM3D_ftox(id) =
GAM3D_stf(id,RegionNum)*0.5*(1.0 + dtanh(stox-GAM3D_khst(id,RegionNum)))
                        End Select
                  End Do
            End If
            !
            !
            !Production
            Do id = 1, ngamcs
                  !
```

```
                  !Ambrose Bug Fix
                  If(GAM3D_stMethod(id,RegionNum) == 1) Then
                        If(UseLOTTMethod == 1) Then
                           AlgaeGrwthNetLimit =
(dmin1(GAM3D_fn(id),GAM3D_fi(id),GAM3D_ft(id),GAM3D_ftox(id)))
                        Else
                           AlgaeGrwthNetLimit =
GAM3D_fn(id)*GAM3D_fi(id)*GAM3D_ft(id)*GAM3D_ftox(id)
                        End if
                        GAM3D_gp(id)          =
GAM3D_k1c(id,RegionNum)*AlgaeGrwthNetLimit
                  Else If(GAM3D_stMethod(id,RegionNum) == 2) Then
                        AlgaeGrwthNetLimit    =
GAM3D_fn(id)*GAM3D_fi(id)*GAM3D_ft(id)
                        GAM3D_gp(id)          =
GAM3D_k1c(id,RegionNum)*AlgaeGrwthNetLimit - GAM3D_ftox(id)
                  End If

                  GAM3D_pnh3(id) =  c(n,k,I_NH3)*c(n,k,I_NO3)/((Epsilon +
GAM3D_khn(id,RegionNum) + c(n,k,I_NH3))*(Epsilon +
GAM3D_khn(id,RegionNum) + c(n,k,I_NO3))) + &

c(n,k,I_NH3)*GAM3D_khn(id,RegionNum)/(Epsilon + (c(n,k,I_NH3) +
c(n,k,I_NO3))*(GAM3D_khn(id,RegionNum) + c(n,k,I_NO3)))

                  GAM3D_dr(id)   =
GAM3D_k1r(id,RegionNum)*GAM3D_Tht_k1r(id, RegionNum)**Temp20
                  GAM3D_dd(id)   =  GAM3D_k1d(id,RegionNum)
!*GAM3D_Tht_k1d(id, RegionNum)**Temp20
                  !
                  !GP,VSK: 04-06-2010
                  GAM_3D_fn(m1(i,j),k,id)      = GAM3D_fn(id)
                  !light limitation factor for GAM in cell n layer k
                  GAM_3D_fi(m1(i,j),k,id)      = GAM3D_fi(id)
                  GAM_3D_ft(m1(i,j),k,id)      = GAM3D_ft(id)
                  GAM_3D_ftox(m1(i,j),k,id)    = GAM3D_ftox(id)
                  GAM_3D_fnet(m1(i,j),k,id)    = AlgaeGrwthNetLimit
                  !N limitation factor for GAM in cell n layer k
                  GAM_3D_N_lim(m1(i,j),k,id) = GAM3D_nl(id)
                  !P limitation factor for GAM in cell n layer k
                  GAM_3D_P_lim(m1(i,j),k,id) = GAM3D_pl(id)
                  !Temperature limitation and effect factor for GAM in
cell n layer k
                  !Ambrose Bug Fix
                  GAM_3D_T_lim(m1(i,j),k,id) = GAM3D_ft(id)
            End Do
            !
            !Estimate each term in the algal rate equation
            Do id = 1, ngamcs
                  Call GAMZooplanktonGrazing
                  MAlgGrowth   =   GAM3D_gp(id)*c(n,k,I_GAM(id))
                  MAlgResp     = - GAM3D_dr(id)*c(n,k,I_GAM(id))
                  MAlgDeath    = - GAM3D_dd(id)*c(n,k,I_GAM(id))
```

```
                MAlgExrec    = -
GAM3D_fe(id,RegionNum)*GAM3D_gp(id)*c(n,k,I_GAM(id))
                MAlgGraze    = - GAM3D_ggraze(id)
                MAlgSettl    = -
GAM3D_ws(id,RegionNum)*c(n,k,I_GAM(id))/d1 + &
                                GAM3D_ws(id,RegionNum)*c(n,k-
1,I_GAM(id))*SetTop/dzk(k-1)
                h(n,k,I_GAM(id)) = h(n,k,I_GAM(id)) + (MAlgGrowth +
MAlgResp + MAlgDeath + MAlgExrec + MAlgGraze + MAlgSettl)*CellVolume
                !
                !GP, VSK: 05-03-2010
                GAM_3D_PProd(m1(i,j),k,id) = MAlgGrowth + MAlgResp +
MAlgDeath + MAlgExrec
                !NH3
                NH3FromAlg   =   GAM3D_dr(id)*anc*c(n,k,I_GAM(id))
                NH3ToAlg     =
GAM3D_gp(id)*anc*GAM3D_pnh3(id)*c(n,k,I_GAM(id))
                h(n,k,I_NH3) =   h(n,k,I_NH3) + (NH3FromAlg -
NH3ToAlg)*CellVolume
                !
                !NO3
                NO3ToAlg     =   GAM3D_gp(id)*anc*(1.0 -
GAM3D_pnh3(id))*c(n,k,I_GAM(id))
                h(n,k,I_NO3) =   h(n,k,I_NO3) - NO3ToAlg*CellVolume
                !
                !PO4
                PO4FromAlg   =   GAM3D_dr(id)*apc*c(n,k,I_GAM(id))
                PO4ToAlg     =   GAM3D_gp(id)*apc*c(n,k,I_GAM(id))

                h(n,k,I_PO4) =   h(n,k,I_PO4) + (PO4FromAlg -
PO4ToAlg)*CellVolume
                !
                !Ambrose Bug Fix
              If(UseLOTTMethod == 1) Then
                !
                     !use the original LOTT study method
                 DOFromAlg    =   GAM3D_gp(id)*(32./12. + &
                                        0.8*32./12.*(1. -
GAM3D_pnh3(id)) + &
                                  0.2*32./12.*(
GAM3D_pnh3(id)))*c(n,k,I_GAM(id))
              Else
                !use the WASP EUTRO method adapted from WASP
DISSOXYG.FOR
                !note: phyto biomass is in mgC/L = gC/m^3
                !growth of phytoplankton using CO2 and NH3
                 DOFromAlg =                  GAM3D_pnh3(id)  *
GAM3D_gp(id) * c(n,k,I_GAM(id)) * 32. / 12.
                !growth of phytoplankton using CO2 and NO3 (2NO3 = 2NH3
+ 3O2)
                 DOFromAlg = DOFromAlg + (1. - GAM3D_pnh3(id)) *
GAM3D_gp(id) * c(n,k,I_GAM(id)) * 32. * (1./12. + 1.5 * anc / 14.)
                End If
```

```
                  DOToAlg      =   32./12.*GAM3D_dr(id)*c(n,k,I_GAM(id))

                  h(n,k,I_DO)  = h(n,k,I_DO) + (DOFromAlg -
DOToAlg)*CellVolume

                  !
                  !WQCBM
                  If(iwqc == WQSSTERMS_USING_WQCBM) Then
                        FCBODFromAlg =  GAM3D_dd(id)*aoc*GAM3D_foc(id,
RegionNum)*GAM3D_fd5(id,RegionNum)*c(n,k,I_GAM(id)) + &

GAM3D_gp(id)*aoc*GAM3D_fe(id,RegionNum)

                        h(n,k,I_CBOD_F) = h(n,k,I_CBOD_F) +
FCBODFromAlg*CellVolume
                        SCBODFromAlg =
GAM3D_dd(id)*aoc*GAM3D_foc(id,RegionNum)*(1 -
GAM3D_fd5(id,RegionNum))*c(n,k,I_GAM(id))

                        h(n,k,I_CBOD_S) = h(n,k,I_CBOD_S) +
SCBODFromAlg*CellVolume
                        !
                        !Ambrose Bug Fix
                        POCFFromAlg  =  GAM3D_dd(id)*fd9f*(1 -
GAM3D_foc(id,RegionNum))*c(n,k,I_GAM(id))  + &
                                        GAM3D_ggraze(id)*fg9f

                        h(n,k,I_OC_P_F) = h(n,k,I_OC_P_F) +
POCFFromAlg*CellVolume

                        POCSFromAlg  =  GAM3D_dd(id)*fd9s*(1 -
GAM3D_foc(id,RegionNum))*c(n,k,I_GAM(id))   + &
                                           GAM3D_ggraze(id)*fg9s

                        h(n,k,I_OC_P_S) = h(n,k,I_OC_P_S) +
POCSFromAlg*CellVolume

                        POCRFromAlg =   GAM3D_dd(id)*fd9r*(1 -
GAM3D_foc(id,RegionNum))*c(n,k,I_GAM(id))   + &
                                         GAM3D_ggraze(id)*fg9r

                        h(n,k,I_OC_P_R) = h(n,k,I_OC_P_R) +
POCRFromAlg*CellVolume
                        !
                        !Adding source to ON_D and OP_D from algae death
and excretion
                        DONFromAlg =
      GAM3D_dd(id)*anc*GAM3D_fon(id,RegionNum)*c(n,k,I_GAM(id)) + &

      GAM3D_gp(id)*anc*GAM3D_fe(id,RegionNum)*c(n,k,I_GAM(id))

                        h(n,k,I_ON_D) = h(n,k,I_ON_D) +
DONFromAlg*CellVolume
```

```
                        DOPFromAlg   =
GAM3D_dd(id)*apc*GAM3D_fop(id,RegionNum)*c(n,k,I_GAM(id))  + &

      GAM3D_gp(id)*apc*GAM3D_fe(id,RegionNum)*c(n,k,I_GAM(id))

                        h(n,k,I_OP_D) =  h(n,k,I_OP_D) +
DOPFromAlg*CellVolume
                        !
                        !Adding source to ON_P and OP_P from algae death
and excretion
                        !Ambrose Bug Fix
                        PONFromAlg   =  GAM3D_dd(id)*anc*(1.0 -
GAM3D_fon(id,RegionNum))*c(n,k,I_GAM(id)) + &
                                        GAM3D_ggraze(id)*anc
                        !
                        ![gN/m3-day]  =       [1/day] * [gN/gC] * [] *
[gC/m3]
                        !               + [gC/m3/day] * [gN/gC] * [] *
[gC/m3]

                        h(n,k,I_ON_P) = h(n,k,I_ON_P) +
PONFromAlg*CellVolume
                        ![gN/day]    =    [gN/day]   +  [gN/m3-day]  *
[m3]
                        !
                        !Ambrose Bug Fix
                        POPFromAlg    = GAM3D_dd(id)*apc*(1 -
GAM3D_fop(id,RegionNum))*c(n,k,I_GAM(id)) + &
                                        GAM3D_ggraze(id)*apc
                        !
                        ![gP/m3-day]  =      [1/day] * [gP/gC] * [] *
[gC/m3]
                        !              + [gC/m3/day] * [gP/gC] * [] *
[gC/m3]

                        h(n,k,I_OP_P) =  h(n,k,I_OP_P) +
POPFromAlg*CellVolume
                        ![gP/day]     =     [gP/day]   [gP/m3-day]  *
[m3]

                End If
                !
                !WQDPM
                If(iwqc == WQSSTERMS_USING_WQDPM) Then

                        DONFromAlg    =
GAM3D_dd(id)*anc*GAM3D_fon(id,RegionNum)*c(n,k,I_GAM(id)) + &

      GAM3D_gp(id)*anc*GAM3D_fe(id,RegionNum)*c(n,k,I_GAM(id))
                        !
                        ![gN/m3-day]  =   [1/day] * [gN/gC] * [] * [gC/m3]
                        !               + [1/day] * [gN/gC] * [] * [gC/m3]
```

```
                        h(n,k,I_ON_D) = h(n,k,I_ON_D) +
DONFromAlg*CellVolume
                        !
                        ![gN/day]     =    [gN/day]   +  [gN/m3-day]  *
[m3]

                        !
                        !Ambrose Bug Fix
                        PONFromAlg    =  GAM3D_dd(id)*anc*(1.0 -
GAM3D_fon(id,RegionNum))*c(n,k,I_GAM(id)) + &
                                                GAM3D_ggraze(id)*anc
                        ![gN/m3-day]  =       [1/day] * [gN/gC] * [] *
[gC/m3]
                        !               + [gC/m3/day] * [gN/gC] * [] *
[gC/m3]

                        h(n,k,I_ON_P) = h(n,k,I_ON_P) +
PONFromAlg*CellVolume
                        !
                        ![gN/day]    =    [gN/day]   +  [gN/m3-day]  *
[m3]

                        DOPFromAlg   =
GAM3D_dd(id)*apc*GAM3D_fop(id,RegionNum)*c(n,k,I_GAM(id))  + &

      GAM3D_gp(id)*apc*GAM3D_fe(id,RegionNum)*c(n,k,I_GAM(id))
                        !
                        ![gP/m3-day]  =  [1/day] * [gP/gC] * [] * [gC/m3]
                        !              + [1/day] * [gP/gC] * [] * [gC/m3]

                        h(n,k,I_OP_D) =  h(n,k,I_OP_D) +
DOPFromAlg*CellVolume
                        !
                        ![gP/day]     =  [gP/day]   +  [gP/m3-day]  *
[m3]

                        !
                        !Ambrose Bug Fix
                        POPFromAlg    = GAM3D_dd(id)*apc*(1 -
GAM3D_fop(id,RegionNum))*c(n,k,I_GAM(id)) + &
                                                GAM3D_ggraze(id)*apc
                        !
                        ![gP/m3-day]  =       [1/day] * [gP/gC] * [] *
[gC/m3]
                    !               + [gC/m3/day] * [gP/gC] * [] *
[gC/m3]


                        h(n,k,I_OP_P) =  h(n,k,I_OP_P) +
POPFromAlg*CellVolume
                        ![gP/day]     =    [gP/day]   +  [gP/m3-day]  *
[m3]

                End If
```

```
                !
                !WQADD3D
                !Call WQADD3D before GAM
                If(AddWaterQuality) Then
                      h(n,k,I_LDOM) = h(n,k,I_LDOM) + ((1.0 -
Pam)*MAlgDeath + MAlgExrec)*CellVolume
                      !
                      !Ambrose bug fix
                      !h(n,k,I_RDOM) = h(n,k,I_RDOM) + (
Pam)*MAlgDeath*CellVolume
                      h(n,k,I_LPOM) = h(n,k,I_LPOM) + (
Pam)*MAlgDeath*CellVolume
                      !
                      !gm/sec
                      h(n,k,I_ALKL) = acc(I_ALKL)*(h(n,k,I_ALKL) + &
                                                              (  ralkan
*MAlgGrowth*(1.0 - prefam)*50000.0 &
                                                               - ralkaa
*MAlgGrowth*(      prefam)*50000.0 &
                                                               + ralkaa
*MAlgResp*50000.0)*CellVolume)
                      If(WQADD3DA_cnss(I_TIOC-
ncWQADDSt+1,WQADD3DRegionStatus(i1(n),j1(n))+1) == 1) Then
                            !
                            !Sources and sinks due to Phytoplankton
                            h(n,k,I_TIOC) = acc(I_TIOC)*(h(n,k,I_TIOC) +
(rcca*MAlgResp - rcca*MAlgGrowth)*CellVolume)
                      End If
                End If
          End Do
     Return

     Entry GAMZooplanktonGrazing
          !
          !Zooplankton Grazing
          Select Case(GAM3D_ZPGMode(id,RegionNum))
                !
                !g/m^3-day
                Case(ConstantGrazing)
                      GAM3D_gmicro(id) =
GAM3D_kgmicro(id,RegionNum)*GAM3D_Tht_kgmicro(id,RegionNum)**temp20
                      GAM3D_gmacro(id) =
GAM3D_kgmacro(id,RegionNum)*GAM3D_Tht_kgmacro(id,RegionNum)**temp20
                      GAM3D_ggraze(id) = GAM3D_kgmicro(id,RegionNum) +
GAM3D_kgmacro(id,RegionNum)
                !
                !1/day
                !grazing function by Boatman (10/17/97)
                Case(LinearGrazing)
                      GAM3D_gmicro(id) =
GAM3D_kgmicro(id,RegionNum)*GAM3D_Tht_kgmicro(id,RegionNum)**temp20*c(n,k
,I_GAM(id))
```

```
                        GAM3D_gmacro(id) =
GAM3D_kgmacro(id,RegionNum)*GAM3D_Tht_kgmacro(id,RegionNum)**temp20*c(n,k
,I_GAM(id))
                        GAM3D_ggraze(id) = GAM3D_gmicro(id) +
GAM3D_gmacro(id)
                  !
                  !(m^3/g-day)
                  !density dependent
                  Case(DensityDependentGrazing)
                        GAM3D_gmicro(id) =
GAM3D_kgmicro(id,RegionNum)*c(n,k,I_GAM(id))*GAM3D_Tht_kgmicro(id,RegionN
um)**temp20*c(n,k,I_GAM(id))
                        GAM3D_gmacro(id) =
GAM3D_kgmacro(id,RegionNum)*c(n,k,I_GAM(id))*GAM3D_Tht_kgmacro(id,RegionN
um)**temp20*c(n,k,I_GAM(id))
                        GAM3D_ggraze(id) = GAM3D_gmicro(id) +
GAM3D_gmacro(id)
            End Select

      Return


      Entry OpenGAM3DTVRCFiles

            NumTVRCFiles = GAM3D_NumTVRCFiles
            Allocate(GAM3D_nTVRCclms(NumTVRCFiles), Stat = AllocError)
            Allocate(GAM3D_iTVRCamp(NumTVRCFiles,256),
GAM3D_iTVRCcol(NumTVRCFiles,256), GAM3D_iTVRCunit(NumTVRCFiles,256), Stat
= AllocError)
            Allocate(GAM3D_TVRCValue(2,256), Stat = AllocError)
            Allocate(GAM3D_TVRCSTime(NumTVRCFiles), Stat = AllocError)
            Allocate(GAM3D_TVRCETime(NumTVRCFiles), Stat = AllocError)
            Allocate(GAM3D_TVRCInterp(NumTVRCFiles), Stat = AllocError)
            Allocate(GAM3D_TVRCMVNumber(NumTVRCFiles), Stat = AllocError)
            Allocate(GAM3D_EndFileCountKDG(NumTVRCFiles), Stat =
AllocError)

            GAM3D_nTVRCclms = 0
            GAM3D_iTVRCamp = 0.0
            GAM3D_iTVRCcol = 0
            GAM3D_TVRCValue = 0.0
            GAM3D_TVRCInterp = 0
            GAM3D_TVRCMVNumber = 999999999.0
            GAM3D_EndFileCountKDG = 0

            Do jdd = 1, NumTVRCFiles
                  If(GAM3D_UseTVRCFile(jdd) == 0) Cycle
                  message = Trim(GAM3D_TVRCFileName(jdd))
                  If(message(1:len_trim(message)-1) == 'No_Data_File')
Cycle
                  file_name = message(1:len_trim(message)-1)
                  Call file_exist(file_name, ierr)
                  If (ierr < 0) Then
```

```
                        message = 'Error in
opening'//file_name(1:len_trim(file_name))//Char(0)
                        iretw =
MessageBox(GetActiveWindow(),message,'GLLVHT
Model'C,MB_ICONERROR.or.MB_OK)
                        Write(NFErr,'(a)') message(1:len_trim(message))
                        GLLVHTError = 1
                        Return
                  End If
                  Write(*,'(a)')
'Initializing'//file_name(1:len_trim(file_name))
                  Call GetUnitNumber(NFGAM3D(jdd))
                  Open (NFGAM3D(jdd), file = file_name, Status='old',
SHARED, Action = 'READ')
                  ReWind(NFGAM3D(jdd))
                  message = ''
                  Read (NFGAM3D(jdd),'(a)') message
                  VersionCheck = 0

                  VersionCheck = 0
                  If(Index(message,'V1') /= 0) VersionCheck = 1
                  If(Index(message,'V2') /= 0) VersionCheck = 2
                  If(Index(message,'V3') /= 0) VersionCheck = 3
                  If(Index(message,'V4') /= 0) VersionCheck = 4
                  Write(*,*) '.KDG TVRC File Number =', jdd, 'Version
Number = ', VersionCheck

                  Do While (.true.)
                        Read (NFGAM3D(jdd),'(a)') message
                        If(Index(message,'$') == 0) Exit
                  End Do
                  BackSpace(NFGAM3D(jdd))
                  If(VersionCheck >= 1) Read(NFGAM3D(jdd),*) Value1,
Value2
                  !Read(NFGAM3D(jdd),*) TVDMVNumber(jdd)


                  Read(NFGAM3D(jdd),*) BytVal1   !use only one bin
                  Read(NFGAM3D(jdd),*) GAM3D_nTVRCclms(jdd)
                  Do i = 1, GAM3D_nTVRCclms(jdd)
                        Read(NFGAM3D(jdd),*) GAM3D_iTVRCCol(jdd,i),
BytVal1, BytVal2, GAM3D_iTVRCAmp(jdd,i), BytVal3, message
                        If(GAM3D_iTVRCCol(jdd,i) == SkipTVDColumnValue)
Cycle
                        GAM3D_iTVRCUnit(jdd,i) = BytVal2
                  End Do
                  !
                  !
                  Do While (.true.)
                        Read (NFGAM3D(jdd),'(a)') message
                        If(index(message,'$') == 0) Exit
                  End Do
                  Backspace(NFGAM3D(jdd))
                  !
```

```
                  !Bin was removed on 08-27-2007 VSK
                  Read(NFGAM3D(jdd),*,iostat = istat) tvdsyear, tvdsmonth,
tvdsday, tvdshour, tvdsmin,&

(GAM3D_TVRCValue(1,i), i = 1, GAM3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFGAM3D(jdd),GAM3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) GAM3D_EndFileCountKDG(jdd) =
GAM3D_EndFileCountKDG(jdd) + 1
                  GAM3D_TVRCSTime(jdd) = dttm2julian(tvdsyear, tvdsmonth,
tvdsday, tvdshour, tvdsmin)
                  !
                  !Bin was removed on 08-27-2007 VSK
                  Read(NFGAM3D(jdd),*,iostat = istat) tvdsyear, tvdsmonth,
tvdsday, tvdshour, tvdsmin,&

(GAM3D_TVRCvalue(2,i), i = 1, GAM3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFGAM3D(jdd),GAM3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) GAM3D_EndFileCountKDG(jdd) =
GAM3D_EndFileCountKDG(jdd) + 1
                  BackSpace(NFGAM3D(jdd))
                  GAM3D_TVRCETime(jdd) = dttm2julian(tvdsyear, tvdsmonth,
tvdsday, tvdshour, tvdsmin)

                  If (mdltime < GAM3D_TVRCSTime(jdd)) Then
                        Write(NFlog,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFlog,'(a)') 'File Name:
'//GAM3D_TVRCFileName(jdd)
                        Write(NFlog,'(a)') 'Algae Model'
                        Write(NFlog,'(/)')

                        Write(NFcle,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFcle,'(a)') 'File Name:
'//GAM3D_TVRCFileName(jdd)
                        Write(NFcle,'(a)') 'Algae Model'
                        Write(NFcle,'(/)')
                  End If
            End Do

      Return


      Entry ReadGAM3DTVRCFiles(jd)
```

```
            If(GAM3D_NumTVRCFiles == 0) Return
            If(.Not. GAM3D_UseTVRCFile(jd)) Return

            Do While (.true. .and. EndFileCountWDG(jd) <= 1)
                  !
                  !
                  If(MdlTime >= GAM3D_TVRCSTime(jd) .and. MdlTime <=
GAM3D_TVRCETime(jd)) Then

                        !SP 10/27/2007 added to re-read the values
                        Backspace(NFGAM3D(jd))
                        Read(NFGAM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (GAM3D_TVRCvalue(1,nci), nci = 1, GAM3D_nTVRCclms(jd))

                        Read(NFGAM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (GAM3D_TVRCvalue(2,nci), nci = 1, GAM3D_nTVRCclms(jd))
                        Backspace(NFGAM3D(jd))
                        !End SP 10/27/2007 added to re-read the values

                        Factor = 0.0
                        Select Case(GAM3D_TVRCInterp(jd))
                              Case(0)
                                    Factor = 0.0
                              Case(1)
                                    Factor = (MdlTime -
GAM3D_TVRCSTime(jd))/(GAM3D_TVRCETime(jd) - GAM3D_TVRCSTime(jd))
                              Case(2)
                                    Factor = 1.0
                        End Select
                        !
                        !Linear interpolation in both depth and time
                        Do nci = 1,GAM3D_NTVRCclms(jd)
                              Value1 = GAM3D_TVRCvalue(1,nci)
                              Value2 = GAM3D_TVRCvalue(2,nci)
                              If(Value1 /= GAM3D_TVRCMVNumber(jd) .and.
Value2 /= GAM3D_TVRCMVNumber(jd)) Then
                                    Value = Factor*Value2 + (1.0 -
Factor)*Value1
                              Else If(Value1 == GAM3D_TVRCMVNumber(jd)
.and. Value2 /= GAM3D_TVRCMVNumber(jd)) Then
                                    Value = Value2
                              Else If(Value1 /= GAM3D_TVRCMVNumber(jd)
.and. Value2 == GAM3D_TVRCMVNumber(jd)) Then
                                    Value = Value1
                              End If
                              If(GAM3D_iTVRCcol(jd,nci) ==
SkipTVDColumnValue) Cycle
                              nj = GAM3D_iTVRCcol(jd,nci)/(ngamcs +
Epsilon)
                              ni = GAM3D_iTVRCcol(jd,nci) - nj*ngamcs + 1
                              Select Case(nj)
```

```
                                    Case(1)
                                          GAM3D_k1r(ni,RegionNum)
            = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(2)
                                          GAM3D_Tht_k1r(ni,RegionNum)
      = Value
                                    Case(3)
                                          GAM3D_k1c(ni,RegionNum)
            = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(4)
                                          GAM3D_Tht_k1c(ni,RegionNum)
      = Value
                                    Case(5)
                                          GAM3D_k1d(ni,RegionNum)
            = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(6)
                                          GAM3D_Tht_k1d(ni,RegionNum)
      = Value
                                    Case(7)
                                          GAM3D_fe(ni,RegionNum)
      = Value
                                    Case(8)
                                          GAM3D_as(ni,RegionNum)
      = Value
                                    Case(9)
                                          GAM3D_ws(ni,RegionNum)
      = Value
                                    Case(10)
                                          GAM3D_ZPGMode(ni,RegionNum)
      = Value
                                    Case(11)
                                          GAM3D_kgmicro(ni,RegionNum)
      = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(12)
                                          GAM3D_Tht_kgmicro(ni,RegionNum) =
Value
                                    Case(13)
                                          GAM3D_kgmacro(ni,RegionNum)
      = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(14)
                                          GAM3D_kgmacro(ni,RegionNum)
      = Value
                                    Case(15)
                                          GAM3D_Tht_kgmacro(ni,RegionNum) =
Value
                                    Case(16)
                                          GAM3D_cchl(ni,RegionNum)
      = Value
                                    Case(17)
                                          GAM3D_LightModel(ni,RegionNum)
      = Value
                                    Case(19)
                                          GAM3D_kke(ni,RegionNum)
            = Value
```

```
                                    Case(20)
                                          GAM3D_kechl(ni,RegionNum)
      = Value
                                    Case(21)
                                          GAM3D_ISat(ni,RegionNum)
      = LightConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(22)
                                          GAM3D_khn(ni,RegionNum)
            = Value
                                    Case(23)
                                          GAM3D_khp(ni,RegionNum)
            = Value
                                    Case(24)
                                          GAM3D_stMethod(ni,RegionNum) =
Value
                                    Case(25)
                                          GAM3D_stf(ni,RegionNum)
            = RateConversion(Value, GAM3D_iTVRCUnit(jd,nci))
                                    Case(26)
                                          GAM3D_khst(ni,RegionNum)
      = Value
                                    Case(27)
                                          GAM3D_tm(ni,RegionNum)
      = Value
                                    Case(28)
                                          GAM3D_ktg1(ni,RegionNum)
      = Value
                                    Case(29)
                                          GAM3D_ktg2(ni,RegionNum)
      = Value
                                    Case(30)
                                          GAM3D_fd5(ni,RegionNum)
            = Value
                                    Case(31)
                                          GAM3D_fon(ni,RegionNum)
            = Value
                                    Case(32)
                                          GAM3D_fop(ni,RegionNum)
            = Value
                                    Case(33)
                                          GAM3D_foc(ni,RegionNum)
            = Value
                              End Select
                        End Do
                        Exit
                  Else
                        !
                        !vsk 01/04/00
                        If(GAM3D_TVRCSTime(jd) > MdlTime) Return
                        Read(NFGAM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                 (GAM3D_TVRCvalue(1,nci), nci = 1, GAM3D_nTVRCclms(jd))
```

```
                        Call
seterrormessage(istat,NFGAM3D(jd),GAM3D_TVRCFileName(jd))
                        If(istat < 0) GAM3D_EndFileCountKDG(jd) =
GAM3D_EndFileCountKDG(jd) + 1
                        If(istat > 0) then
                              GLLVHTError = 1
                              Return
                        End If

                        GAM3D_TVRCSTime(jd) = dfloat(dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin))

                        Read(NFGAM3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                 (GAM3D_TVRCvalue(2,nci), nci = 1, GAM3D_nTVRCclms(jd))
                        Backspace(NFGAM3D(jd))
                        Call
seterrormessage(istat,NFGAM3D(jd),GAM3D_TVRCFileName(jd))
                        If(istat < 0) GAM3D_EndFileCountKDG(jd) =
GAM3D_EndFileCountKDG(jd) + 1
                        If(istat > 0) Then
                              GLLVHTError = 1
                              Return
                        End If
                        GAM3D_TVRCETime(jd) = dfloat(dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin))
                   End If
              End Do  !jd

      Return


    Entry WriteGAM3DParametersToSnapShot

            Write(NFSnp,'(/)')
          Write(NFSnp,'(a)') &

      '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@'
            Write(NFSnp,'(a)') &
            '                                GEMSS-GAM
'
            Write(NFSnp,'(a)') &
            '           Generalized Algae Model Kinetics Rates and
Constants         '
          Write(NFSnp,'(a)') &

      '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@'

            Do j = 1, NumGAM3DRegions + 1
                  Write(NFSnp,'(/)')
                  Write(NFSnp,'(a)') &
```

```
      '*****************************************************************
******'
                  If(j == 1) Then
                        Write(NFSnp,'(a)') 'Region: Whole Waterbody'
                  Else
                        Write(NFSnp,'(a,a)') 'Region: ',
GAM3DRegionName(j)
                  End If
                  Write(NFSnp,'(a)') &

      '*****************************************************************
******'
                  Write(NFSnp,'(/)')
                  Do i = 1, ngamcs
                        message = ''
                        Write(NFSnp,'(a)') &

'#################################################################################
'
                        message = 'Rates and Constants For '
                        Write(message,'(a,a)') &
                        message(1:len_trim(message)),
cnm(i)(1:len_trim(cnm(I_GAM(id))))
                        Write(NFSnp,'(a)') message(1:len_trim(message))

                        Write(NFSnp,'(a)') &

      '#################################################################
######'

                        If(GAM3D_UseNutrientLimit(j) == 1) Then
                        Write(NFSnp,'(a,a,a)') &
                        'Use nurtient limit
=', ' Yes',' '
                        Else
                        Write(NFSnp,'(a,a,a)') &
                        'Use nurtient limit
=', ' No',' '
                        End If

                        If(GAM3D_UseTempLimit(j) == 1) Then
                        Write(NFSnp,'(a,a,a)') &
                        'Use temperature limit
=', ' Yes',' '
                        Else
                        Write(NFSnp,'(a,a,a)') &
                        'Use temperature limit
=', ' No',' '
                        End If

                        If(GAM3D_UseSalineToxicLimit(j) == 1) Then
                        Write(NFSnp,'(a,a,a)') &
```

```
                      'Use salinity limit
=', ' Yes',' '
                      Else
                      Write(NFSnp,'(a,a,a)') &
                      'Use salinity limit
=', ' No',' '
                      End If

                      If(GAM3D_UseLightLimit(j) == 1) Then
                      Write(NFSnp,'(a,a,a)') &
                      'Use light limit
=', ' Yes',' '
                      Else
                      Write(NFSnp,'(a,a,a)') &
                      'Use light limit
=', ' No',' '
                      End If

                      Write(NFSnp,'(a,e13.6,a)') &
                      'k1r,  Resipiration rate at 20 C
=', GAM3D_k1r(i,j),  ' 1/sec'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'Tht_k1r,  Temperature coeffcient for k1r
=', GAM3D_Tht_k1r(i,j),  ' '

                      Write(NFSnp,'(a,e13.6,a)') &
                      'k1c,  Growth rate at 20 C
=', GAM3D_k1c(i,j),  ' 1/sec'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'Tht_k1c,  Temperature coeffcient for k1c
=', GAM3D_Tht_k1c(i,j),  ' '

                      Write(NFSnp,'(a,e13.6,a)') &
                      'k1d,  Growth rate at 20 C
=', GAM3D_k1d(i,j),  ' 1/sec'

                      Write(NFSnp,'(a,e13.6,a)') &
                      'fe,  Excretion fraction
=', GAM3D_fe(i,j),  ' '
                      Write(NFSnp,'(a,e13.6,a)') &
                      'as,  Assimilation efficiency of zooplankton
grazing                         =', GAM3D_as(i,j), ' '

                      Write(NFSnp,'(a,e13.6,a)') &
                      'ws,  Settling velocity
=', GAM3D_ws(i,j), ' m/sec'
                      !
                      !
                      Select Case(GAM3D_ZPGMode(i,j))
                            !
                            !
                           Case(1)
                      Write(NFSnp,'(a,a13,a)') &
```

```
                        'Zooplankton grazing type
=', ' Constant',' '
                              !
                              !
                              Case(2)
                        Write(NFSnp,'(a,a13,a)') &
                        'Zooplankton grazing type
=', ' Linear',' '

                              !
                              Case(3)
                        Write(NFSnp,'(a,a16,a)') &
                        'Zooplankton grazing type
=', ' Density Dependent',' '
                        End Select

                        Write(NFSnp,'(a,e13.6,a)') &
                        'kgmicro, Graing rate due to zooplankton
=', GAM3D_kgmicro(i,j),' 1/sec'

                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_kgmicro, Temperature coefficient for
zooplankton grazing rate           =', GAM3D_Tht_kgmicro(i,j),' '

                        Write(NFSnp,'(a,e13.6,a)') &
                        'kgmacro, Graing rate due to zooplankton
=', GAM3D_kgmacro(i,j),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_kgmacro, Temperature coefficient for
zooplankton grazing rate           =', GAM3D_Tht_kgmacro(i,j),' '

                        Write(NFSnp,'(a,e13.6,a)') &
                        'cchl, Carbon to chlorophyll ratio
=', GAM3D_cchl(i,j),' gC/gChl-a'

                        Select Case(GAM3D_LightModel(i,j))
                              !
                              !
                              Case(1)
                        Write(NFSnp,'(a,a13,a)') &
                        'Light model used for algal growth
=', 'Half Saturation', ' '
                              !
                              Case(2)
                        Write(NFSnp,'(a,a13,a)') &
                        'Light model used for algal growth
=', 'Full Saturation', ' '
                              !
                              Case(3)
                        Write(NFSnp,'(a,a13,a)') &
                        'Light model used for algal growth
=', 'No Saturation', ' '
                        End Select
```

```
                      Write(NFSnp,'(a,e13.6,a)') &
                      'kke, Light extinction coefficient
=', GAM3D_kke(i,j), '  '

                      Write(NFSnp,'(a,e13.6,a)') &
                      'kechl, Light attenuation coefficient
=', GAM3D_kechl(i,j), ' m^2/mg'

                      Write(NFSnp,'(a,e13.6,a)') &
                      'ISat, Light constant
=', GAM3D_ISat(i,j), ' W/m^2',' '

                      Write(NFSnp,'(a,e13.6,a)') &
                      'khn, Constant for algae nitrogen uptake
=', GAM3D_khn(i,j), ' gm N/m^3'

                      Write(NFSnp,'(a,e13.6,a)') &
                      'khp, Constant for algae phosphorous uptake
=', GAM3D_khp(i,j), ' gm P/m^3'
                      !
                      !Salinity Toxicity
                      Select Case(GAM3D_stMethod(i,j))
                            !
                            !
                            Case(1)
                      Write(NFSnp,'(a,a13,a)') &
                      'Salinity toxicity is computed using
=', ' Equation 1','  '
                            !
                            !
                            Case(2)
                      Write(NFSnp,'(a,a13,a)') &
                      'Salinity toxicity is computed using
=', ' Equation 2','  '
                      End Select
                      !
                      !
                      Write(NFSnp,'(a,e13.6,a)') &
                      'stf, Maximum mortality due to salinity toxicity
=', GAM3D_stf(i,j), ' 1/sec'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'khst, Salinity at which toxicity is half the
maximum value                =', GAM3D_khst(i,j), ' ppt'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'tm, Optimum temperature for algae growth
=', GAM3D_tm(i,j), ' 1/sec'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'ktg1, Suboptimal temperature effect for algal
growth                       =', GAM3D_ktg1(i,j), ' C'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'ktg2, Superoptimal temperature effect for algal
growth                     =', GAM3D_ktg2(i,j), ' C'
                      Write(NFSnp,'(a,e13.6,a)') &
```

```
                        'fd5, Fraction of dead phytoplankton recycled to
fast CBOD                     =', GAM3D_fd5(i,j), ' '

                        Write(NFSnp,'(a,e13.6,a)') &
                        'fon, Organic nitrogen from dead algae
=', GAM3D_fon(i,j), ' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'fop, Organic phosphorus from dead aglage
=', GAM3D_fop(i,j), ' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'foc, Organic carbon from dead algae
=', GAM3D_foc(i,j), ' '

                End Do
            End Do

      Return

End Subroutine GAMSubs
```

WQADD Module

```
Subroutine WQADD
      Use dfwin
      Use G7allocmemory
      Use AllocMemoryWQCBM3DVariables
      Use AllocMemoryWQADD3DVariables
      Use GEMSS_MGMAllocMemoryVariables
      Include 'G7Variables.f90'
      Include 'G7WQADD3DVariables.f90'
      Include 'G7WQCBMVariables.f90'
      !
      !Local variables used in different subroutines
      Integer(2)  NumTVRCFiles
      Integer(2)  i, j, k, jd, nci, nc, jdd, rdd
      Integer(4)   n
      Integer(4)  ierr, iRetW, iStat, DTTM2JULIAN
      Real(8)           d1, BotLight, Depth, Term1, Term2
      Real(8)           FBnb, FBpb, Phi_nt, Phi_ps, Phi_nb, Phic, Phi_Lb,
Phi_Sb
      Real(8)           BotAlgPhoto, BotAlgResp, BotAlgExrec, BotAlgDeath
      Real(8)           DONFromBotAlg, BotAlgUptakeN, PLim, BotAlgUptakeP,
SbN, SbP
      Real(8)           Pab, NH3ToBotAlg, NO3ToBotAlg, DOPFromBotAlg,
IOPToBotAlg
      Real(8)           DOFromBotAlg, DOToBotAlg, Value1, Value2, Factor,
Value
      Byte        BytVal1, BytVal2, BytVal3, BotAlgModelingMethod, Units,
WQADD3D_UseTVRCFileLocal, VersionCheck
      Logical(1)  ChangesInRegions, ReadDefaultSettingsOnce
      Integer(2)   VariableCount, TVRCFileCount
      Character(256) message1, message2, WQADD3D_TVRCFileNameLocal
      Real(8)           RateConversion, SolRadInLPerDay,
VelocityConversion, DiameterConversion
      !
      !Local input rates and constants variables for Macrophyte
      Real(8)           mgC, mgN, mgP, mgD, mgA, Cgb20, gC, aoc
      Real(8)           ThtCgb20, abMax, qN, qP
      Real(8)           keb, Thtkeb, kdb, Thtkdb, krb, Thtkrb, ksPb, ksNb,
ksCb, pCO2
      Real(8)           NUpWCFactor, PUpWCFactor
      Byte        BotAlgLightModel, BotAlgGrowthModelType
      Real(8)           KLb, q0N, q0P, rmN, rmP, KqN, KqP, CgbT
      Real(8)           BotAlgRefTempForCgb, BotAlgRefTempkrb,
BotAlgRefTempkeb, BotAlgRefTempkbd
      Real(8)           krbT, kebT, kdbT
      Real(8)           Foxb, Ksob, NLim, khnxb
      Byte        IkoxB, HCO3UseF, pHSolveMethod, pCO2SetMethod
      !
      !Local variables for Alkalinity, pH and Conductivity
      Real(8) ralkaa, ralkan, ralkbn, ralkbp, ralkn, ralkden, rondn
      Real(8) rcca, rcco, rccd, rccc, ralkda, ralkdn, defa
      Real(8) ph_k1, ph_k2, ph_kw, ph_kh
      Real(8) alp0, alp1, alp2, denom
      Real(8)     kacT, anc, apc, ron, roa, aca, ph_hh
      Real(8) CO2Sat, CO2, HCO3, CO3, TIOC, ALKL,ACF
```

```
      Real(8) prefam
      !
      !
      Real(8) ancp, apcp

      !SP 05/16/2008
      Real(8) FCBODFromBotAlg, SCBODFromBotAlg, POCFFromBotAlg,
POCSFromBotAlg
      Real(8) POCRFromBotAlg, PONFromBotAlg, POPFromBotAlg
      Real(8) foc, fd5, fd9f, fd9s, fd9r, fon, fop
      Real(8) foc_WQADD, fd5_WQADD, fd9f_WQADD, fd9s_WQADD, fd9r_WQADD,
fon_WQADD, fop_WQADD
      !End SP 05/16/2008
      !
      !Organic matter variables
      Real(8) K_LDOM, Tht_K_LDOM, K_L2RDOMTR, K_RDOM, Tht_K_RDOM
      Real(8) K_L2RPOMTR, K_LPOM, Tht_K_LPOM, K_RPOM, Tht_K_RPOM
      Real(8) kds_LPOM, Tht_kds_LPOM, kds_RPOM, Tht_kds_RPOM, vs_LPOM,
vs_RPOM
      Real(8) LDOMDecay, L2RDOMTransfer, RDOMDecay
      Real(8) LPOMDecay, L2RPOMTransfer, LPOMSettling, RPOMDecay,
RPOMSettling
      Real(8) LPOMDissolution, RPOMDissolution
      Real(8) r_OMC, r_OMP, r_OMN, Pam
      Real(8) Tht_K_L2RPOMTR, Tht_K_L2RDOMTR

      Real(8) Nu, Sgr, dks, Settle_TSS
      !
      !Variables for pHCO2 code from CE-QUAL-W2
      Real(8) cart, alkt, t1k, sqrs2, s2
      Real(8) dh1, dh2, h2co3t, hco3t, co3t, oh
      Real(8) akw, ak1, ak2, pht, incr, fph,hion, bicart
      Integer(2) Iter, ni

      Real(8)          aoc1


      Data BotAlgRefTempForCgb, BotAlgRefTempkrb, BotAlgRefTempkeb,
BotAlgRefTempkbd/&
                           20.0,             20.0,             20.0,
20.0/
      Data ReadDefaultSettingsOnce/.True./

      !Data Epsilon /1.0d-30/
      Common/WQADD3DLocalVariables/&
                  NumTVRCFiles,&
                  nci,&
                  ierr, iRetW, iStat,&
                  BotLight, Depth,&
                  FBnb, FBpb, Phi_nt, Phi_ps, Phi_nb, Phic, Phi_Lb,
Phi_Sb,&
                  BotAlgPhoto, BotAlgResp, BotAlgExrec, BotAlgDeath,&
                  DONFromBotAlg, BotAlgUptakeN, PLim, BotAlgUptakeP, SbN,
SbP,&
```

```
                  Pab, NH3ToBotAlg, NO3ToBotAlg, DOPFromBotAlg,
IOPToBotAlg,&
                  DOFromBotAlg, DOToBotAlg, Value1, Value2, Factor,
Value,&
                  BytVal1, BytVal2, BytVal3, BotAlgModelingMethod, CO2,
CO3, HCO3, ALKL, TIOC

      Common/WQADD3DRCLocalVariables/&
                  mgC, mgN, mgP, mgD, mgA, Cgb20, gC,&
                  ThtCgb20, abMax, qN, qP,&
                  keb, Thtkeb, kdb, Thtkdb, krb, Thtkrb, ksPb, ksNb,
ksCb,pCO2,&
                  NUpWCFactor, PUpWCFactor,&
                  BotAlgLightModel, BotAlgGrowthModelType,&
                  KLb, q0N, q0P, rmN, rmP, KqN, KqP, CgbT,&
                  BotAlgRefTempForCgb, BotAlgRefTempkrb, BotAlgRefTempkeb,
BotAlgRefTempkbd,&
                  krbT, kebT, kdbT,&
                  Foxb, Ksob, NLim, khnxb,&
                  IkoxB, HCO3UseF, pHSolveMethod, pCO2SetMethod, &
                  K_LDOM, Tht_K_LDOM, K_L2RDOMTR, K_RDOM, Tht_K_RDOM,&
                  K_L2RPOMTR, K_LPOM, Tht_K_LPOM, K_RPOM, Tht_K_RPOM,&
                  kds_LPOM, Tht_kds_LPOM, kds_RPOM, Tht_kds_RPOM, vs_LPOM,
vs_RPOM,&
                  LDOMDecay, L2RDOMTransfer, RDOMDecay,&
                  LPOMDecay, L2RPOMTransfer, LPOMSettling, RPOMDecay,
RPOMSettling,&
                  LPOMDissolution, RPOMDissolution,&
                  r_OMC, r_OMP, r_OMN, apc, anc, ancp, apcp, aoc, &
                  Tht_K_L2RPOMTR, Tht_K_L2RDOMTR
                  !
                  !
      Common/CommonVariablesForAllWQModels/&
                  i, j, k, n, nc, d1, aoc1, &
                  foc_WQADD, fd5_WQADD, fd9f_WQADD, fd9s_WQADD,
fd9r_WQADD, fon_WQADD, fop_WQADD
      !
      !pH
      Common/CommonVariablesForpH/&
                  ralkaa, ralkan, ralkbn, ralkbp, ralkn, ralkden, rondn,&
                  rcca, rcco, rccd, rccc, ralkda, ralkdn, defa,&
                  ph_k1, ph_k2, ph_kw, ph_kh,&
                  alp0, alp1,&
                  kacT, ron, roa, aca, ph_hh,&
                  CO2Sat,&
                  prefam
      !
      !GAM
      Common/CommonVariablesBetweenGAMandWQADD3D/&
                  Pam
      !
      !SSS
      Common/CommonVariablesForSSS/&
                  Nu, Sgr, dks
```

```
      Save

      Entry GetWQADD3DCoefficients

            If(NumWQADD3DRegions /= 0) Then
                  If(WQADD3DRegionStatus(i,j) /= 0 .and.
WQADD3DRegionRCDStatus(WQADD3DRegionStatus(i,j)) == 1) Then
                        Call SetWQADD3DRatesAndConstantsLocalChange
                        If(WQADD3D_NumTVRCFiles > 1) jdd =
WQADD3DRegionStatus(i,j) + 1
                  Else
                        Call SetWQADD3DRatesAndConstantsLocalDefault
                        jdd = 1
                  End If
            Else
                  If(ReadDefaultSettingsOnce) Then
                        Call SetWQADD3DRatesAndConstantsLocalDefault
                        ReadDefaultSettingsOnce = .False.
                        jdd = 1
                  End If
            End If

            Call ReadWQADD3DTVRCFiles(jdd)

      Return
      !
      !
      Entry WQADD3DSSTerms

            aoc = aoc1

            !SP 05/16/2008
            foc = foc_WQADD;       fd5 = fd5_WQADD;       fd9f =
fd9f_WQADD;      fd9s = fd9s_WQADD;
            fd9r = fd9r_WQADD;           fon = fon_WQADD;       fop =
fop_WQADD
            !End SP 05/16/2008
            !
            !Bottom Layer
            Call GetWQADD3DCoefficients
            BotAlgUptakeN = 0.0d+00
            BotAlgUptakeP = 0.0d+00
            !SP 05/27/2008
            rdd = 1
            If(NumWQADD3DRegions /= 0)rdd = WQADD3DRegionStatus(i,j)+1
            If(k == k0(i,j) .and. acc(I_MPHYT) == 1 .and.
WQADD3DA_cnss(I_MPHYT-ncWQADDSt+1,rdd) == 1) Call
IntegrationForBottomAlgae
            If((acc(I_pH) == 1 .and. acc(I_TIOC) == 1 .and. acc(I_ALKL)
== 1 .and. acc(I_COND) == 1).or. acc(I_COND) == 1) Call pHCONDAndTIOC
            If(acc(I_LDOM) == 1 .or. acc(I_RPOM) == 1 .or. acc(I_LPOM) ==
1 .or. acc(I_RPOM) == 1) Call OrganicMatter

      Return
```

```
      Entry WQADDSSSKinetics

            If(acc(I_SSS) == 0) Return
            If(TSSVelType == 1) Call ComputeSettlingVelcityWQDPM
            Settle_TSS      = - (TSSVel*c(n,k,I_SSS)/d1
       &          !TSS settling
                                      +  TSSVel*SetTop*c(n,k-
1,I_SSS)/dzk(k-1))
            h(n,k,I_SSS)     =     (h(n,k,I_SSS) + WQADD3DA_cnss(I_SSS-
ncWQADDSt+1,WQADD3DRegionStatus(i,j)+1)*Settle_TSS*CellVolume)

      Return
      !
      !Do this inside water quality models
      Entry pHCONDAndTIOC


            TIOC = c(n,k,I_TIOC)/(12000.0)  ! to convert from gm/m^3 to
Moles/l
            ALKL = c(n,k,I_ALKL)

            If(pHSolveMethod == Newton_Raphson) Then
                  Call pHSolvTest(c(n,k,I_PH), TIOC, c(n,k,I_Temp), ALKL,
c(n,k,I_COND),NFLog)
            Else
                  Call pHSolve(c(n,k,I_PH), TIOC, c(n,k,I_Temp), ALKL,
c(n,k,I_COND),NFLog)
            End If
            Call ChemRates(c(n,k,I_Temp), ph_k1, ph_k2, ph_kw, ph_kh,
c(n,k,I_COND))
            ph_hh = 10.0**(-c(n,k,I_PH))
            !
            !CO2 concentration
            denom = ph_hh**2.0 + ph_k1*ph_hh + ph_k1*ph_k2
            alp0  = ph_hh**2.0/denom
            CO2   = alp0*TIOC
            !
            !Bicaronate concentration
            alp1 = ph_k1*ph_hh/denom
            HCO3 = alp1*TIOC
            !
            !Carbonate concentration
            alp2 = ph_k1*ph_k2/denom      !fraction of TIOC as carbonate
[CO3--]
            CO3  = alp2*TIOC
            !
            !At the water surface
            If(k == kt) Then
                  kacT = (32.0/44.0)**0.25*k2T
                  ACF  = (1.0 - (el(kt) - z(n  ))/1000.0/44.3)**5.25
                  !
                  !Is pCO2 still in units of ppm here? Note that ph_kh is
Henry’s constant calculated at line 208
```

```
                If(pCO2SetMethod == 1) Then
                      CO2Sat = 0.286*exp(-0.0314*c(n,k,I_Temp))*ACF
                Else
                      CO2Sat = kacT*pCO2/ 1000000.0
                End If
                h(n,kt,I_TIOC) = h(n,kt,I_TIOC) !+ kacT*CellArea*(CO2Sat
- CO2)
           End If
           !
           !Phyto and bottom algae
           !SP 05/27/2008
           rdd = 1
           If(NumWQADD3DRegions /= 0)rdd = WQADD3DRegionStatus(i,j)+1
           If(WQADD3DA_cnss(I_TIOC-ncWQADDSt+1,rdd) == 1) Then
                !
                !Sources and sinks due to Phytoplankton
                !h(n,k,I_TIOC) = acc(I_TIOC)*(h(n,k,I_TIOC) +
(rcca*PhytoResp - rcca*PhytoPhoto)*CellVolume)
                h(n,k,I_TIOC) = acc(I_TIOC)*(h(n,k,I_TIOC) + (PhytoResp
- PhytoPhoto)*CellVolume)
                !
                !Sources and sinks due to bottom algae
                !SP 05/27/2008
                !Bob Ambrose: 06-14-2011: CellVolume changed to CellArea
                h(n,k,I_TIOC) = acc(I_TIOC)*(h(n,k,I_TIOC) +
WQADD3DA_cnss(I_SSS-ncWQADDSt+1,rdd)*(rccd*BotAlgResp -
rccd*BotAlgPhoto)*CellArea)

           End If
           !
           !Alkalinity sources and sinks due to phytoplankton and
nitrogen components
           !SP 05/27/2008
           If(WQADD3DA_cnss(I_ALKL-ncWQADDSt+1,rdd) == 1) Then
                If(iwqc == WQSSTERMS_USING_WQE5M .or. iwqc ==
WQSSTERMS_USING_WQDPM) Then
                      prefam = pnh3
                Else If(iwqc == WQSSTERMS_USING_WQCBM) Then
                      prefam = pnh3_d
                End If
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) - ralkaa *
PhytoPhoto * prefam * 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) + ralkan *
PhytoPhoto * (1 - prefam) * 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) + ralkaa *
PhytoResp * 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) - ralkbn *
BotAlgUptakeN * prefamF * 50000 - ralkbp * BotAlgUptakeP * 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) + ralkbn *
BotAlgUptakeN * (1 - prefamF) * 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) - ralkn * NH4Nitrif
* 50000
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) + ralkden * Denitr
* 50000
```

```
                !dc(i, nv - 2, 1) = dc(i, nv - 2, 1) + ralkbn * OrgNHydr
* 50000 + ralkbp * OrgPHydr * 50000
                !
                !gm/sec
                h(n,k,I_ALKL) = acc(I_ALKL)*(h(n,k,I_ALKL) + &
                                                        (  ralkan
*PhytoPhoto*(1.0 - prefam)*50000.0 &
                                                           - ralkaa
*PhytoPhoto*(      prefam)*50000.0 &
                                                           + ralkaa
*PhytoResp*50000.0                 &
                                                           - ralkn
*NH4Nitrif*50000.0                 &
                                                           +
ralkden*DeNitr   *50000.0                  &
                                                           + ralkbn
*OrgNHydr *50000.0                 &
                                                           + ralkbp
*OrgPHydr *50000.0               )*CellVolume)
                !
                !Alkalinity sources and sinks due to bottom algae
                h(n,k,I_ALKL) = acc(I_ALKL)*(h(n,k,I_ALKL) &
                                                          + (-
ralkbn*BotAlgUptakeN*(      Pab)*50000.0 &
                                                          -
ralkbp*BotAlgUptakeP            *50000.0 &
                                                          +
ralkbn*BotAlgUptakeN*(1.0 - Pab)*50000.0)*1.0e-03)  !mgN or mgP/sec to
gm/sec

           End If
      Return
      !
      !Organic matter
      Entry OrganicMatter

           !SP 05/27/2008
           rdd = 1
           If(NumWQADD3DRegions /= 0)rdd = WQADD3DRegionStatus(i,j)+1

           !
           !Labile Particulate Organic Matter
           If(acc(I_LPOM) == 1 .and. WQADD3DA_cnss(I_LPOM-
ncWQADDSt+1,rdd) == 1) Then
                LPOMDecay      = K_LPOM*Tht_K_LPOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_LPOM)
                !
                !GUI Change
                L2RPOMTransfer =
K_L2RPOMTR*Tht_K_L2RPOMTR**(c(n,k,I_Temp) - 20.0)*c(n,k,I_LPOM)
                LPOMSettling   = vs_LPOM*c(n,k,I_LPOM)/d1 -
SetTop*vs_LPOM*c(n,k-1,I_LPOM)/dzk(k-1)
                LPOMDissolution= kds_LPOM*Tht_kds_LPOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_LPOM)
```

```
                h(n,k,I_LPOM)  = h(n,k,I_LPOM) +
(PhytoDeath*Pam*PhytoDandEinWQADD/r_OMC - LPOMDecay - L2RPOMTransfer -
LPOMSettling -  &
                                        LPOMDissolution)*CellVolume
            End If
            !
            !Refractory Particulate Organic Matter
            If(acc(I_RPOM) == 1 .and. WQADD3DA_cnss(I_RPOM-
ncWQADDSt+1,rdd) == 1) Then
                RPOMDecay      = K_RPOM*Tht_K_RPOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_RPOM)
                !Bob Ambrose: 06-14-2011
                !GUI Change
                L2RPOMTransfer =
K_L2RPOMTR*Tht_K_L2RPOMTR**(c(n,k,I_Temp) - 20.0)*c(n,k,I_LPOM)
                RPOMSettling   = vs_RPOM*c(n,k,I_RPOM)/d1 -
SetTop*vs_RPOM*c(n,k-1,I_RPOM)/dzk(k-1)
                RPOMDissolution= kds_RPOM*Tht_kds_RPOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_RPOM)
                h(n,k,I_RPOM)  = h(n,k,I_RPOM) + (L2RPOMTransfer -
RPOMDecay - RPOMSettling - &
                                        RPOMDissolution)*CellVolume
            End If

            !SP 11/16/2011 Moved down to add Dissolution source
            !
            !Labile Dissolved Organic Matter
            If(acc(I_LDOM) == 1 .and. WQADD3DA_cnss(I_LDOM-
ncWQADDSt+1,rdd) == 1) Then
                LDOMDecay      = K_LDOM*Tht_K_LDOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_LDOM)
                !GUI Change
                L2RDOMTransfer =
K_L2RDOMTR*Tht_K_L2RDOMTR**(c(n,k,I_Temp) - 20.0)*c(n,k,I_LDOM)
                h(n,k,I_LDOM)  = h(n,k,I_LDOM) + (PhytoDeath*(1.0 -
Pam)*PhytoDandEinWQADD/r_OMC + PhytoExcretion*PhytoDandEinWQADD/r_OMC - &
                                     LDOMDecay - L2RDOMTransfer +
LPOMDissolution)*CellVolume
            End If
            !
            !Refractory Dissolved Organic Matter
            If(acc(I_RDOM) == 1 .and. WQADD3DA_cnss(I_RDOM-
ncWQADDSt+1,rdd) == 1) Then
                RDOMDecay      = K_RDOM*Tht_K_RDOM**(c(n,k,I_Temp) -
20.0)*c(n,k,I_RDOM)
                h(n,k,I_RDOM)  = h(n,k,I_RDOM) + (L2RDOMTransfer -
RDOMDecay + RPOMDissolution)*CellVolume
            End If

            !
            !OC, PO4 and NH3 sources and sinks from OM
            !SP 08/31/2011 Skip OM decay if CBOD constituents are active
```

```
            If(acc(I_TIOC) == 1 .and. WQADD3DA_cnss(I_TIOC-
ncWQADDSt+1,rdd) == 1)Then

                  h(n,k,I_TIOC) = h(n,k,I_TIOC) + (CBODDecay*r_CBODC*(1-
PhytoDandEinWQADD))*CellVolume
                  h(n,k,I_TIOC) = h(n,k,I_TIOC) + (LDOMDecay*r_OMC +
RDOMDecay*r_OMC + LPOMDecay*r_OMC +
RPOMDecay*r_OMC)*PhytoDandEinWQADD*CellVolume


            End If
            !
            !Phosphate OM sources
            If(acc(I_PO4) == 1) h(n,k,I_PO4)  =  (h(n,k,I_PO4)  +
acc(I_PO4)*(LDOMDecay*r_OMP + RDOMDecay*r_OMP &
                                                                  +
LPOMDecay*r_OMP + RPOMDecay*r_OMP)*CellVolume)
            !
            !Ammonium OM sources
            If(acc(I_NH3) == 1) h(n,k,I_NH3)  =  (h(n,k,I_NH3)  +
acc(I_NH3)*(LDOMDecay*r_OMN + RDOMDecay*r_OMN &
                                                                   +
LPOMDecay*r_OMN + RPOMDecay*r_OMN)*CellVolume)
      Return

      Entry IntegrationForBottomAlgae
            !
            !
            nc = I_MPHYT
            Select Case(BotAlgModelingMethod)
                  !
                  !Euler
                  Case(1)
                        Call BottomAlgae
                        c(n,k,nc) = c0(n,k,nc) + dt*h(n,k,nc)
                        INb(n) = INb(n) + dt*dINb(n)
                        IPb(n) = IPb(n) + dt*dIPb(n)
                        If(INb(n) < 0) INb(n) = Epsilon
                        If(IPb(n) < 0) IPb(n) = Epsilon
                        If(c(n,k,nc) < 0) c(n,k,nc) = Epsilon
                  !
                  !Runge Kutta method
                  Case(2)
                        !
                        !
                        Call BottomAlgae

                        INbOld(n) = INb(n)
                        IPbOld(n) = IPb(n)
                        c(n,k,nc) = c(n,k,nc) + (dt/2.0)*h(n,k,nc)
                        INb(n) = INb(n) + (dt/2.0)*dINb(n)
                        IPb(n) = IPb(n) + (dt/2.0)*dIPb(n)
                        h(n,k,nc) = 0.0d+00
```

```
                        Call BottomAlgae

                        c(n,k,nc) = c0(n,k,nc) + (dt/2.0)*h(n,k,nc)
                        dc2(n) = h(n,k,nc)
                        INb(n) = INb(n) + (dt/2.0)*dINb(n)
                        IPb(n) = IPb(n) + (dt/2.0)*dIPb(n)
                        h(n,k,nc) = 0.0d+00

                        Call BottomAlgae
                        dINb3(n) = dINb(n)
                        dIPb3(n) = dIPb(n)
                        c(n,k,nc) = c0(n,k,nc) + (dt/2.0)*h(n,k,nc)
                        dc3(n) = h(n,k,nc)
                        dINb(n) = dINb(n) + (dt/2.0)*dINb(n)
                        dIPb(n) = dIPb(n) + (dt/2.0)*dIPb(n)
                        h(n,k,nc) = 0.0d+00

                        Call BottomAlgae
                        dINb4(n) = dINb(n)
                        dIPb4(n) = dIPb(n)
                        c(n,k,nc) = c0(n,k,nc) + (dt/6.0)*(dc1(n) +
2.0*(dc2(n) + dc3(n) + dc4(n)))
                        INb(n) = INbOld(n) + (dt/6.0)*(dINb(n) +
2.0*(dINbm(n) + dINb3(n) + dINb4(n)))
                        IPb(n) = IPbOld(n) + (dt/6.0)*(dIPb(n) +
2.0*(dIPbm(n) + dIPb3(n) + dIPb4(n)))

            End Select


      Return

      Entry BottomAlgae

            !SP 04/24/2008 Conversion of KqN and KqP
            !KqN coming in as dimensionless
            !q0N is in mgN/gD
            !So KqN becomes in mgN/gD
            KqN = KqN*q0N
            !KqP coming in as dimensionless
            !q0P is in mgP/gD
            !So KqP becomes in mgP/gD
            KqP = KqP*q0P
            !End SP 04/24/2008 Conversion KqN and KqP

            Select Case (IkoxB)
                  Case (1)
                        Foxb = c(n,k,I_DO) / (Ksob + c(n,k,I_DO))
                  Case (2)
                        Foxb = 1 - Exp(-Ksob * c(n,k,I_DO))
                  Case (3)
                        Foxb = c(n,k,I_DO)**2.0/(Ksob + c(n, k,
I_DO)**2.0)
            End Select
```

```
            !
            !Attenuation coeffcient from WQM models
            ke = LEC(n,k)
            CgbT = Cgb20*ThtCgb20**(c(n,k,I_Temp) - BotAlgRefTempForCgb)
!gD/m^2-sec

            !qN in mgN/gD
            !INb in mgN/m2
            If(c(n,k,I_MPHYT) > 0) Then
                  qN = INb(n)/c(n,k,I_MPHYT)  !concentration in gD/m^2
                  qP = IPb(n)/c(n,k,I_MPHYT)
            Else
                  qN = ana/ada
                  qP = apa/ada
            End If
            !
            !Nitrogen
            If(qN > 0) Then
                  FBnb = q0N/qN
            Else
                  FBnb = 1.0
            End If
            If (FBnb < 0) FBnb = 0
            If (FBnb > 1) FBnb = 1
            !
            !Phosphorous
            If(qP > 0) Then
                  FBpb = q0P/qP
            Else
                  FBpb = 1
            End If

            If(FBpb < 0) FBpb = 0
            If(FBpb > 1) FBpb = 1

            Phi_nt = 1.0 - FBnb
            Phi_ps = 1.0 - FBpb
            Phi_nb = dmin1(Phi_nt, Phi_ps)
            !
            !Term3 is used if pH constituent is used in the water quality
model
            If (hco3useF == 0) Then
            !     !CO is the limiting substrate
                  phic = alp0*c(n,k,I_TIOC)/(kscb + alp0*c(n,k,I_TIOC))
            Else
            !     !HCO3 is the limiting substrate
                  phic = (alp0 + alp1)*c(n,k,I_TIOC)/(kscb + (alp0 +
alp1)*c(n,k,I_TIOC))
            End If
            phic = 9999.0
            Phi_nb = dmin1(Phi_nb,phic)
            !
            !Light Limitation
```

```
            If(WaterQuality) Then
                  BotLight = RBotPAR(n,k0(i,j))
            Else
                  SolRadInLPerDay =
PAR*SolRad/((4.183076*100*100)/86400.0)  !Cal/cm^2-day
                  BotLight = SolRadInLPerDay*exp(-ke*Depth)
            End If

            Select Case(BotAlgLightModel)
                  !
                  !Half-Saturation
                  Case(1)
                        If(KLb + BotLight <= 0) Then
                              phi_Lb = 1
                        Else
                              Phi_Lb = BotLight/(KLb + BotLight)
                        End If
                  !
                  !Smith
                  Case(2)
                        If(BotLight**2.0 + KLb**2.0 == 0) Then
                              Phi_Lb = 1
                        Else
                              Phi_Lb = BotLight/sqrt(KLb**2.0 +
BotLight**2.0)
                        End If
                  !
                  !Steele
                  Case(3)
                        If (KLb <= 0) Then
                              Phi_Lb = 1
                        Else
                              Phi_Lb = (BotLight/KLb)*exp(1.0 -
BotLight/KLb)
                        End If
            End Select
            !
            !Space Limitation
            Phi_Sb = 1.0 - c(n,k,I_MPHYT)/abMax

            If(BotAlgGrowthModelType == 1) Then
                  !
                  !For this CgbT is zero order photosynthesis rate
gD/(m^2-sec)
                  BotAlgPhoto = CgbT*Phi_nb*Phi_Lb
            Else
                  !
                  !For this, CgbT is photosynthesis rate 1/sec
                  BotAlgPhoto = CgbT*Phi_nb*Phi_lb*Phi_sb*c(n,k,I_MPHYT)
            End If
            !
            !Respiration - gD/m^2-sec
            krbT = krb*Thtkrb**(c(n,k,I_Temp) - BotAlgRefTempkrb)
            BotAlgResp = Foxb*krbT*c(n,k,I_MPHYT)
```

```
            !
            !Excretion - gD/m^2-sec
            kebT = keb*Thtkeb**(c(n,k,I_Temp) - BotAlgRefTempkeb)
            BotAlgExrec =kebT*c(n,k,I_MPHYT)
            !
            !Death - gD/m^2-sec
            kdbT = kdb*Thtkdb**(c(n,k,I_Temp) - BotAlgRefTempkbd)
            BotAlgDeath = kdbT*c(n,k,I_MPHYT)
            !
            !gD/m^2-sec
            !SP 05/16/2008
            h(n,k,I_MPHYT) = h(n,k,I_MPHYT) + (BotAlgPhoto - BotAlgResp -
BotAlgDeath)
            !End SP 05/16/2008
            !
            !Dissolved Organic Nitrogen Source - Equation 118
            !(mgN/gD)x(gD/m^2-sec)x(m^2) - mgN/sec
            !SP 05/16/2008               DONFromBotAlg =
qN*BotAlgDeath*CellArea

            If((ksNb + c(n,k,I_NH3) + c(n,k,I_NO3)) > 0.00 ) Then
                  NLim = KqN/(KqN + (qN - q0N))
                  If (NLim < 0) NLim = 0
                  If (NLim > 1) NLim = 1
                  !
                  !Units are in mgN/sec
                  BotAlgUptakeN =
CellArea*rMN*c(n,k,I_MPHYT)*((c(n,k,I_NH3) + c(n,k,I_NO3))/(ksNb +
c(n,k,I_NH3) + c(n,k,I_NO3)))*nLim
            Else
                  BotAlgUptakeN = 0.0
            End If
            !
            !Dissolved Organic Phosphorous Source
            If(ksPb + c(n,k,I_PO4) > 0) Then
                  PLim = (KqP/(KqP+ qP - q0P))
                  If (PLim < 0) PLim = 0
                  If (PLim > 1) PLim = 1
                  !Units are in mgP/sec
                  BotAlgUptakeP =
CellArea*rmP*c(n,k,I_MPHYT)*(c(n,k,I_PO4)/(ksPb + c(n,k,I_PO4)))*PLim
            Else
                  BotAlgUptakeP = 0.0
            End If

            !
            !mgN/m^2-sec - change in intracellular nitrogen concentration
            dINb(n) = (BotAlgUptakeN - CellArea*qN*BotAlgDeath -
CellArea*qN*BotAlgExrec)/CellArea
            !
            !mgP/m^2-sec - change in intracellular phosphorus
concentration
            dIPb(n) = (BotAlgUptakeP - CellArea*qP*BotAlgDeath -
CellArea*qP*BotAlgExrec)/CellArea
```

```
            !
            !Ammonia Nitrogen Sink - Equation 121
            Term1 = (c(n,k,I_NH3)*c(n,k,I_NO3))/((khnxb        +
c(n,k,I_NH3))*(khnxb + c(n,k,I_NO3)))
            Term2 = (c(n,k,I_NH3)*khnxb)       /((c(n,k,I_NH3) +
c(n,k,I_NO3) + Epsilon)*(khnxb + c(n,k,I_NO3)))
            Pab   = Term1 + Term2
            !
            !mgN/sec
            NH3ToBotAlg = Pab*BotAlgUptakeN*NUpWCFactor
            !
            !Nitrate Nitrogen Sink - Equation 128 - mgN/sec
            NO3ToBotAlg = (1.0 - Pab)*BotAlgUptakeN*NUpWCFactor
            !
            !Dissolved Organic Phosphorous Source - Equation 130
            !(mgN/gD)X(gD/m^2-sec) = mgN/m^2-sec
            !SP 05/16/2008              DOPFromBotAlg =
qP*BotAlgDeath*CellArea
            !
            !Inorganic Phosphorous Sink - Equation 133
            !mgP/sec
            IOPToBotAlg = BotAlgUptakeP*PUpWCFactor
            !
            !Dissolved Oxygen Source - Equation 136
            !mgO2/m^2-sec -> mgO2/sec
            !gO2/s
            DOFromBotAlg = (rod*BotAlgPhoto*Pab + rod * BotAlgPhoto * (1
- Pab) * 138 / 107)*CellArea
            !
            !Dissolved Oxygen Sink due to bottom algae respiration -
Equation 136
            !mgO2/m^2-sec -> gmO2/sec
            !gO2/s
            DOToBotAlg = rod*BotAlgResp*CellArea

            !SP 05/16/2008
            FCBODFromBotAlg       =     BotAlgDeath*rod*foc*fd5*CellArea
            SCBODFromBotAlg       =     BotAlgDeath*rod*foc* (1-
fd5)*CellArea
            POCFFromBotAlg        =     BotAlgDeath*(1/adc)*(1-
foc)*fd9f*CellArea
            POCSFromBotAlg        =     BotAlgDeath*(1/adc)* (1-
foc)*fd9s*CellArea
            POCRFromBotAlg        =     BotAlgDeath *(1/adc)*(1-
foc)*fd9r*CellArea
            DONFromBotAlg         =     BotAlgDeath*qN* fon*CellArea+
BotAlgExrec*qN*CellArea
            DOPFromBotAlg         =     BotAlgDeath*qP* fop*CellArea+
BotAlgExrec*qP*CellArea
            PONFromBotAlg         =     BotAlgDeath*qN*(1-fon)*CellArea
            POPFromBotAlg         =     BotAlgDeath*qP*(1-fop)*CellArea
            !End SP 05/16/2008

            WQADD_3D_nt(n) = Phi_nt
```

```
            WQADD_3D_ps(n) = Phi_ps
            WQADD_3D_Lb(n) = phi_Lb
            WQADD_3D_Sb(n) = phi_Sb
            WQADD_3D_nb(n) = Phi_nb

            Select Case(iwqc)
                  !
                  !WQE5M
                  Case(WQSSTERMS_USING_WQE5M)
                        h(n,k,I_ON)  = acc(I_ON)*(h(n,k,I_ON)   +
DONFromBotAlg*1.0e-03)
                        h(n,k,I_OP)  = acc(I_OP)*(h(n,k,I_OP)   +
DOPFromBotAlg*1.0e-03)
                        h(n,k,I_NH3) = acc(I_NH3)*(h(n,k,I_NH3) -
NH3ToBotAlg*1.0e-03)
                        h(n,k,I_NO3) = acc(I_NO3)*(h(n,k,I_NO3) -
NO3ToBotAlg*1.0e-03)
                        h(n,k,I_PO4) = acc(I_PO4)*(h(n,k,I_PO4) -
IOPToBotAlg*1.0e-03)
                        h(n,k,I_DO)  = acc(I_DO)*(h(n,k,I_DO)   +
(DOFromBotAlg - DOToBotAlg))
                        !prefam = pnh3
                  !
                  !WQDPM
                  Case(WQSSTERMS_USING_WQDPM)
                        h(n,k,I_ON_D)  = acc(I_ON_D)*(h(n,k,I_ON_D)     +
DONFromBotAlg*1.0e-03)
                        h(n,k,I_OP_D)  = acc(I_OP_D)*(h(n,k,I_OP_D)     +
DOPFromBotAlg*1.0e-03)
                        h(n,k,I_NH3) = acc(I_NH3)*(h(n,k,I_NH3)       -
NH3ToBotAlg*1.0e-03)
                        h(n,k,I_NO3) = acc(I_NO3)*(h(n,k,I_NO3)       -
NO3ToBotAlg*1.0e-03)
                        h(n,k,I_PO4) = acc(I_PO4)*(h(n,k,I_PO4)       -
IOPToBotAlg*1.0e-03)
                        h(n,k,I_DO)  = acc(I_DO)*(h(n,k,I_DO)         +
(DOFromBotAlg - DOToBotAlg))
                        !prefam = pnh3
                  !
                  !WQCBM
                  Case(WQSSTERMS_USING_WQCBM)
                        h(n,k,I_ON_D)    = acc(I_ON_D)*(h(n,k,I_ON_D) +
DONFromBotAlg*1.0e-03)
                        h(n,k,I_OP_D)    = acc(I_OP_D)*(h(n,k,I_OP_D) +
DOPFromBotAlg*1.0e-03)
                        !SP 05/16/2008
                        h(n,k,I_CBOD_F) = acc(I_CBOD_F)*h(n,k,I_CBOD_F)
      + FCBODFromBotAlg
                        h(n,k,I_CBOD_S) = acc(I_CBOD_S)*h(n,k,I_CBOD_S)
      + SCBODFromBotAlg
                        h(n,k,I_OC_P_F) = acc(I_OC_P_F)*h(n,k,I_OC_P_F)
      + POCFFromBotAlg
                        h(n,k,I_OC_P_S) = acc(I_OC_P_S)*h(n,k,I_OC_P_S)
      + POCSFromBotAlg
```

```
                h(n,k,I_OC_P_R) = acc(I_OC_P_R)*h(n,k,I_OC_P_R)
+ POCRFromBotAlg
                h(n,k,I_ON_P)    = acc(I_ON_P)*h(n,k,I_ON_P)
+ PONFromBotAlg *1.0e-03
                h(n,k,I_OP_P)    = acc(I_OP_P)*h(n,k,I_OP_P)
+ POPFromBotAlg *1.0e-03
                !End SP 05/16/2008
                h(n,k,I_NH3)     = acc(I_NH3)*(h(n,k,I_NH3)
- NH3ToBotAlg*1.0e-03)
                h(n,k,I_NO3)     = acc(I_NO3)*(h(n,k,I_NO3)
- NO3ToBotAlg*1.0e-03)
                h(n,k,I_PO4)     = acc(I_PO4)*(h(n,k,I_PO4)
- IOPToBotAlg*1.0e-03)
                h(n,k,I_DO)      = acc(I_DO)*(h(n,k,I_DO)
+ (DOFromBotAlg - DOToBotAlg))
                !prefam = pnh3_d
      End Select

Return


Entry SetWQADD3DRatesAndConstantsLocalDefault

      mgC      = WQADD3D_mgC
      mgN      = WQADD3D_mgN
      mgP      = WQADD3D_mgP
      mgD      = WQADD3D_mgD
      mgA      = WQADD3D_mgA
      Cgb20    = WQADD3D_Cgb20
      ThtCgb20   = WQADD3D_ThtCgb20
      BotAlgGrowthModelType = WQADD3D_BotAlgGrowthModelType
      abMax    = WQADD3D_abMax
      krb      = WQADD3D_krb
      Thtkrb   = WQADD3D_Thtkrb
      keb      = WQADD3D_keb
      Thtkeb   = WQADD3D_Thtkeb
      kdb      = WQADD3D_kdb
      Thtkdb   = WQADD3D_Thtkdb
      ksPb     = WQADD3D_ksPb
      ksNb     = WQADD3D_ksNb
      ksCb     = WQADD3D_ksCb
      BotAlgLightModel = WQADD3D_BotAlgLightModel
      PAR          = WQADD3D_PAR
      KLb      = WQADD3D_KLb
      khnxb    = WQADD3D_khnxb
      q0N      = WQADD3D_q0N
      q0P      = WQADD3D_q0P
      rmN      = WQADD3D_rmN
      rmP      = WQADD3D_rmP
      KqN      = WQADD3D_KqN
      KqP      = WQADD3D_KqP
      IkoxB  = WQADD3D_IkoxB
      Ksob     = WQADD3D_Ksob
      BotAlgModelingMethod = WQADD3D_BotAlgModelingMethod
```

```
      HCO3UseF = WQADD3D_HCO3UseF

      pCO2SetMethod = WQADD3D_pCO2SetMethod
      pCO2   = WQADD3D_pCO2
      pHSolveMethod = WQADD3D_pHSolveMethod
      !
      !Organic Matter
      K_LDOM            = WQADD3D_K_LDOM
      Tht_K_LDOM   = WQADD3D_Tht_K_LDOM

      K_L2RDOMTR   = WQADD3D_K_L2RDOMTR
      Tht_K_L2RDOMTR = WQADD3D_Tht_K_L2RDOMTR

      K_RDOM            = WQADD3D_K_RDOM
      Tht_K_RDOM   = WQADD3D_Tht_K_RDOM

      K_LPOM            = WQADD3D_K_LPOM
      Tht_K_LPOM   = WQADD3D_Tht_K_LPOM

      K_L2RPOMTR   = WQADD3D_K_L2RPOMTR
      Tht_K_L2RPOMTR = WQADD3D_Tht_K_L2RPOMTR

      vs_LPOM      = WQADD3D_vs_LPOM
      kds_LPOM     = WQADD3D_kds_LPOM
      Tht_kds_LPOM = WQADD3D_Tht_kds_LPOM

      vs_RPOM           = WQADD3D_vs_RPOM
      K_RPOM            = WQADD3D_K_RPOM
      Tht_K_RPOM   = WQADD3D_Tht_K_RPOM

      kds_RPOM     = WQADD3D_kds_RPOM
      Tht_kds_RPOM = WQADD3D_Tht_kds_RPOM


      r_OMN        = WQADD3D_r_OMN
      r_OMP        = WQADD3D_r_OMP
      r_OMC        = WQADD3D_r_OMC

      Pam               = WQADD3D_Pam

      PUpWCFactor = WQADD3D_PUpWCFactor
      NUpWCFactor = WQADD3D_NUpWCFactor

      TSSVelType  = WQADD3D_TSSVelType
      TSSVel      = WQADD3D_TSSVel
      TSSSize     = WQADD3D_TSSSize
      TSSDensity  = WQADD3D_TSSDensity

      Call SetLocalDataUnitConversion

Return


Entry SetWQADD3DRatesAndConstantsLocalChange
```

```
            RegionNum = WQADD3DRegionStatus(i,j)
            If(RegionNum == 0) Return
            mgC      = WQADD3DA_mgC(RegionNum)
            mgN      = WQADD3DA_mgN(RegionNum)
            mgP      = WQADD3DA_mgP(RegionNum)
            mgD      = WQADD3DA_mgD(RegionNum)
            mgA      = WQADD3DA_mgA(RegionNum)
            Cgb20    = WQADD3DA_Cgb20(RegionNum)
            ThtCgb20 = WQADD3DA_ThtCgb20(RegionNum)

            BotAlgGrowthModelType = WQADD3D_BotAlgGrowthModelType

            abMax    = WQADD3DA_abMax(RegionNum)
            krb      = WQADD3DA_krb(RegionNum)
            Thtkrb   = WQADD3DA_Thtkrb(RegionNum)
            keb      = WQADD3DA_keb(RegionNum)
            Thtkeb   = WQADD3DA_Thtkeb(RegionNum)
            kdb      = WQADD3DA_kdb(RegionNum)
            Thtkdb   = WQADD3DA_Thtkdb(RegionNum)
            ksPb     = WQADD3DA_ksPb(RegionNum)
            ksNb     = WQADD3DA_ksNb(RegionNum)
            ksCb     = WQADD3DA_ksCb(RegionNum)

            BotAlgLightModel = WQADD3DA_BotAlgLightModel(RegionNum)

            PAR          = WQADD3DA_PAR(RegionNum)
            KLb      = WQADD3DA_KLb(RegionNum)
            khnxb    = WQADD3DA_khnxb(RegionNum)
            q0N      = WQADD3DA_q0N(RegionNum)
            q0P      = WQADD3DA_q0P(RegionNum)
            rmN      = WQADD3DA_rmN(RegionNum)
            rmP      = WQADD3DA_rmP(RegionNum)
            KqN      = WQADD3DA_KqN(RegionNum)
            KqP      = WQADD3DA_KqP(RegionNum)
            IkoxB  = WQADD3DA_IkoxB(RegionNum)
            Ksob     = WQADD3DA_Ksob(RegionNum)

            BotAlgModelingMethod =
WQADD3DA_BotAlgModelingMethod(RegionNum)

            pCO2SetMethod = WQADD3DA_pCO2SetMethod(RegionNum)
            pCO2   = WQADD3DA_pCO2(RegionNum)

            K_LDOM                = WQADD3DA_K_LDOM(RegionNum)
            Tht_K_LDOM       = WQADD3DA_Tht_K_LDOM(RegionNum)
            K_L2RDOMTR       = WQADD3DA_K_L2RDOMTR(RegionNum)
            Tht_K_L2RDOMTR   = WQADD3DA_Tht_K_L2RDOMTR(RegionNum)

            K_RDOM                = WQADD3DA_K_RDOM(RegionNum)
            Tht_K_RDOM       = WQADD3DA_Tht_K_RDOM(RegionNum)

            K_LPOM                = WQADD3DA_K_LPOM(RegionNum)
            Tht_K_LPOM       = WQADD3DA_Tht_K_LPOM(RegionNum)
```

```
      K_L2RPOMTR       = WQADD3DA_K_L2RPOMTR(RegionNum)
      Tht_K_L2RPOMTR   = WQADD3DA_Tht_K_L2RPOMTR(RegionNum)
      vs_LPOM               = WQADD3DA_vs_LPOM(RegionNum)
      kds_LPOM         = WQADD3DA_kds_LPOM(RegionNum)
      Tht_kds_LPOM     = WQADD3DA_Tht_kds_LPOM(RegionNum)

      K_RPOM                = WQADD3DA_K_RPOM(RegionNum)
      Tht_K_RPOM       = WQAdd3DA_Tht_K_RPOM(RegionNum)
      vs_RPOM               = WQADD3DA_vs_RPOM(RegionNum)
      kds_RPOM         = WQADD3DA_kds_RPOM(RegionNum)
      Tht_kds_RPOM     = WQADD3DA_Tht_kds_RPOM(RegionNum)
      Pam                   = WQADD3DA_Pam(RegionNum)
      !
      !Total suspended solids
      TSSVelType  = WQADD3DA_TSSVelType(RegionNum)
      TSSVel      = WQADD3DA_TSSVel(RegionNum)
      TSSSize     = WQADD3DA_TSSSize(RegionNum)
      TSSDensity  = WQADD3DA_TSSDensity(RegionNum)


      PUpWCFactor = WQADD3DA_PUpWCFactor(RegionNum)
      NUpWCFactor = WQADD3DA_NUpWCFactor(RegionNum)


      Call SetLocalDataUnitConversion

Return


Entry SetLocalDataUnitConversion


      RateValue(1) = mgC
      RateValue(2) = mgN
      RateValue(3) = mgP
      RateValue(4) = mgD
      RateValue(5) = mgA

      gD  = rateValue(4) / 1000
      gC  = rateValue(1) / 1000
      anc = rateValue(2) / gC
      apc = rateValue(3) / gC
      ron = ron / 1000
      adc = gD / gC          !gD/gC
      roa = aoc * gC / rateValue(5)
      ana = rateValue(2) / rateValue(5)
      apa = rateValue(3) / rateValue(5)
      aca = gC / rateValue(5)
      ada = gD / rateValue(5)
      roc = aoc                         !gO2/gC
      rod = roc/adc                     !gO2/gD
      !
      !pH/Inorganic carbon stoichiometry
      ralkaa   = 14.0d+00/106.0d+00/12.0d+00*aca/1000.0d+00
```

```
            ralkan   = 18.0d+00/106.0d+00/12.0d+00*aca/1000.0d+00
            ralkbn   = 16.0d+00/16.0d+00/14.0d+00/1000.0d+00/1000.0d+00
            ralkbp   = 2.0d+00/1.0d+00/31.0d+00/1000.0d+00/1000.0d+00
            ralkn    = 2.0d+00/14.0d+00/1000.0d+00/1000.0d+00
            ralkden  = 4.0d+00/(4.0d+00*14.0+00)/1000.0d+00/1000.0d+00
            rondn    = roc*5.0d+00*12.0d+00/(4.0d+00*14.0d+00)/1000.0d+00
            rcca     = gC/mgA/12.0d+00/1000.0d+00
            rcco     = 1.0d+00/12.0d+00/roc/1000.0d+00
            rccd     = gC/gD/12.0d+00/1000.0d+00
            rccc     = 1.0d+00/12/1000.0d+00
            ralkda   = 14.0d+00/106.0d+00/12.0d+00/adc/1000.0d+00
            ralkdn   = 18.0d+00/106.0d+00/12.0d+00/adc/1000.0d+00
            !
            !Convert bottom algae rates from a mgA to gD basis
            !"Zero-Order"
            If (BotAlgGrowthModelType == 1) Cgb20 = Cgb20*gD/mgA   !from
mgA to gD
            abmax = abmax*gD/mgA        !convert from mgA/m^2 to gD/m^2
            q0N = q0N*mgA/gD             !convert from mgN/mgA to mgN/gD
            q0P = q0P*mgA/gD             !convert from mgP/mgA to mgP/gD
            rmN = rmN*mgA/gD             !convert from mgN/mgA to mgN/gD
            rmP = rmP*mgA/gD             !convert from mgP/mgA to mgP/gD
            !SP 04/24/2008 Conversion moved to main code
            !KqN = KqN*q0N               !convert from unitless to mgN/gD
            !KqP = KqP*q0P               !convert from unitless to mgP/gD

            ksNb  = ksNb*1.0e-03  !convert from ugN/L to mgN/L
            ksPb  = ksPb*1.0e-03  !convert from ugP/L to mgP/L
            khnxb = khnxb*1.0e-03 !convert from ugN/L to mgN/L



      Return


      Entry OpenWQADD3DTVRCFiles


            NumTVRCFiles = WQADD3D_NumTVRCFiles
            Allocate(WQADD3D_nTVRCclms(NumTVRCFiles), Stat = AllocError)
            Allocate(WQADD3D_iTVRCamp(NumTVRCFiles,256),
WQADD3D_iTVRCcol(NumTVRCFiles,256), WQADD3D_iTVRCunit(NumTVRCFiles,256),
Stat = AllocError)
            Allocate(WQADD3D_TVRCValue(2,256), Stat = AllocError)
            Allocate(WQADD3D_TVRCSTime(NumTVRCFiles), Stat = AllocError)
            Allocate(WQADD3D_TVRCETime(NumTVRCFiles), Stat = AllocError)
            Allocate(WQADD3D_TVRCInterp(NumTVRCFiles), Stat = AllocError)
            Allocate(WQADD3D_TVRCMVNumber(NumTVRCFiles), Stat =
AllocError)
            Allocate(WQADD3D_EndFileCountKDG(NumTVRCFiles), Stat =
AllocError)

            WQADD3D_nTVRCclms = 0
            WQADD3D_iTVRCamp = 0.0
```

```
            WQADD3D_iTVRCcol = 0
            WQADD3D_TVRCValue = 0.0
            WQADD3D_TVRCInterp = 0
            WQADD3D_TVRCMVNumber = 999999999.0
            WQADD3D_EndFileCountKDG = 0

            Do jdd = 1, NumTVRCFiles
                  If(WQADD3D_UseTVRCFile(jdd) == 0) Cycle
                  message = Trim(WQADD3D_TVRCFileName(jdd))
                  If(message(1:len_trim(message)-1) == 'No_Data_File')
Cycle
                  file_name = message(1:len_trim(message)-1)
                  Call file_exist(file_name, ierr)
                  If (ierr < 0) Then
                        message = 'Error in
opening'//file_name(1:len_trim(file_name))//Char(0)
                        iretw =
MessageBox(GetActiveWindow(),message,'GLLVHT
Model'C,MB_ICONERROR.or.MB_OK)
                        Write(NFErr,'(a)') message(1:len_trim(message))
                        GLLVHTError = 1
                        Return
                  End If
                  Write(*,'(a)')
'Initializing'//file_name(1:len_trim(file_name))
                  Call GetUnitNumber(NFWQADD3D(jdd))
                  Open (NFWQADD3D(jdd), file = file_name, Status='old',
SHARED, Action = 'READ')
                  ReWind(NFWQADD3D(jdd))
                  message = ''
                  Read (NFWQADD3D(jdd),'(a)') message
                  VersionCheck = 0
                  If(Index(message,'V1') /= 0) VersionCheck = 1
                  If(Index(message,'V2') /= 0) VersionCheck = 2
                  If(Index(message,'V3') /= 0) VersionCheck = 3
                  If(Index(message,'V4') /= 0) VersionCheck = 4
                  Do While (.true.)
                        Read (NFWQADD3D(jdd),'(a)') message
                        If(Index(message,'$') == 0) Exit
                  End Do
                  BackSpace(NFWQADD3D(jdd))

                  If(VersionCheck >= 1) Read(NFWQADD3D(jdd),*) Value1,
Value2
                  Read(NFWQADD3D(jdd),*) BytVal1
                  Read(NFWQADD3D(jdd),*) WQADD3D_nTVRCclms(jdd)
                  Do i = 1, WQADD3D_nTVRCclms(jdd)
                        Read(NFWQADD3D(jdd),*) WQADD3D_iTVRCCol(jdd,i),
BytVal1, BytVal2, WQADD3D_iTVRCAmp(jdd,i), BytVal3, message
                        If(WQADD3D_iTVRCCol(jdd,i) == SkipTVDColumnValue)
Cycle
                        WQADD3D_iTVRCUnit(jdd,i) = BytVal3
                  End Do
                  !
```

```
                  !
                  Do While (.true.)
                        Read (NFWQADD3D(jdd),'(a)') message
                        If(index(message,'$') == 0) Exit
                  End Do
                  Backspace(NFWQADD3D(jdd))
                  Read(NFWQADD3D(jdd),*,iostat = istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&

(WQADD3D_TVRCValue(1,i), i = 1, WQADD3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFWQADD3D(jdd),WQADD3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) WQADD3D_EndFileCountKDG(jdd) =
WQADD3D_EndFileCountKDG(jdd) + 1
                  WQADD3D_TVRCSTime(jdd) = dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin)

                  Read(NFWQADD3D(jdd),*,iostat = istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&

(WQADD3D_TVRCvalue(2,i), i = 1, WQADD3D_nTVRCclms(jdd))
                  Call
seterrormessage(istat,NFWQADD3D(jdd),WQADD3D_TVRCFileName(jdd))
                  If(istat > 0) Then
                        GLLVHTError = 1
                        Return
                  End If
                  If(istat < 0) WQADD3D_EndFileCountKDG(jdd) =
WQADD3D_EndFileCountKDG(jdd) + 1
                  BackSpace(NFWQADD3D(jdd))
                  WQADD3D_TVRCETime(jdd) = dttm2julian(tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin)

                  If (mdltime < WQADD3D_TVRCSTime(jdd)) Then
                        Write(NFlog,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFlog,'(a)') 'File Name:
'//WQADD3D_TVRCFileName(jdd)
                        Write(NFlog,'(a)') 'WQ MOdel : '//'WQADD3D'
                        Write(NFlog,'(/)')

                        Write(NFcle,'(a)') 'TVDS Error: Model start time
is < Rates and Constants TV file start time'
                        Write(NFcle,'(a)') 'File Name:
'//WQADD3D_TVRCFileName(jdd)
                        Write(NFcle,'(a)') 'WQ MOdel : '//'WQADD3D'
                        Write(NFcle,'(/)')
                  End If
            End Do
```

```
      Return


      Entry ReadWQADD3DTVRCFiles(jd)

            If(WQADD3D_NumTVRCFiles == 0) Return
            If(.Not. WQADD3D_UseTVRCFile(jd)) Return

            !SP 10/26/2007 added for SetLocalDataUnitConversion
            RateValue(1) = mgC
            RateValue(2) = mgN
            RateValue(3) = mgP
            RateValue(4) = mgD
            RateValue(5) = mgA

            gD  = rateValue(4) / 1000
            gC  = rateValue(1) / 1000
            anc = rateValue(2) / gC
            apc = rateValue(3) / gC
            ron = ron / 1000
            adc = gD / gC          !gD/gC
            roa = aoc * gC / rateValue(5)
            ana = rateValue(2) / rateValue(5)
            apa = rateValue(3) / rateValue(5)
            aca = gC / rateValue(5)
            ada = gD / rateValue(5)
            roc = aoc                          !gO2/gC
            rod = roc/adc                      !gO2/gD
            !End SP 10/26/2007 added for SetLocalDataUnitConversion

            Do While (.true. .and. WQADD3D_EndFileCountKDG(jd) <= 1)
                  !
                  !
                  If(MdlTime >= WQADD3D_TVRCSTime(jd) .and. MdlTime <=
WQADD3D_TVRCETime(jd)) Then

                        !SP 10/27/2007 added to re-read the values
                        Backspace(NFWQADD3D(jd))
                        Read(NFWQADD3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (WQADD3D_TVRCvalue(1,nci), nci = 1,
WQADD3D_nTVRCclms(jd))

                        Read(NFWQADD3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (WQADD3D_TVRCvalue(2,nci), nci = 1,
WQADD3D_nTVRCclms(jd))
                        Backspace(NFWQADD3D(jd))
                        !End SP 10/27/2007 added to re-read the values

                        Factor = 0.0
                        Select Case(WQADD3D_TVRCInterp(jd))
                              Case(0)
                                    Factor = 0.0
```

```
                              Case(1)
                                    Factor = (MdlTime -
WQADD3D_TVRCSTime(jd))/(WQADD3D_TVRCETime(jd) - WQADD3D_TVRCSTime(jd))
                              Case(2)
                                    Factor = 1.0
                        End Select
                        !
                        !Linear interpolation in both depth and time
                        Do nci = 1,WQADD3D_NTVRCclms(jd)
                              Value1 = WQADD3D_TVRCvalue(1,nci)
                              Value2 = WQADD3D_TVRCvalue(2,nci)
                              If(Value1 /= WQADD3D_TVRCMVNumber(jd) .and.
Value2 /= WQADD3D_TVRCMVNumber(jd)) Then
                                    Value = Factor*Value2 + (1.0 -
Factor)*Value1
                              Else If(Value1 == WQADD3D_TVRCMVNumber(jd)
.and. Value2 /= WQADD3D_TVRCMVNumber(jd)) Then
                                    Value = Value2
                              Else If(Value1 /= WQADD3D_TVRCMVNumber(jd)
.and. Value2 == WQADD3D_TVRCMVNumber(jd)) Then
                                    Value = Value1
                              End If
                              If(WQADD3D_iTVRCcol(jd,nci) ==
SkipTVDColumnValue) Cycle
                              Select Case(WQADD3D_iTVRCcol(jd,nci))
                                    Case(1)
                                          mgC   = Value
                                    Case(2)
                                          mgN = Value
                                    Case(3)
                                          mgP = Value
                                    Case(4)
                                          mgD = Value
                                    Case(5)
                                          mgA = Value
                                    !SP 10/15/2007   Case(6)
                                          !BotAlgGrowthModel = Value
                                    Case(6)
                                          Cgb20 =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                          If (BotAlgGrowthModelType == 1)
Cgb20 = Cgb20*gD/mgA   !from mgA to gD
                                    Case(7)
                                          ThtCgb20 = Value
                                    Case(8)
                                          abMax = Value
                                          abmax = abmax*gD/mgA
!convert from mgA/m^2 to gD/m^2
                                    Case(9)
                                          krb =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(10)
                                          Thtkrb = Value
                                    Case(11)
```

```
                                        keb  =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                  Case(12)
                                        Thtkeb = Value
                                  Case(13)
                                        kdb =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                  Case(14)
                                        Thtkdb = Value
                                  Case(15)
                                        ksNb = Value
                                        ksNb  = ksNb*1.0e-03  !convert
from ugN/L to mgN/L
                                  Case(16)
                                        ksPb = Value
                                        ksPb  = ksPb*1.0e-03  !convert
from ugP/L to mgP/L
                                  Case(17)
                                        ksCb = Value
                                  !SP 10/15/2007   Case(18)
                                        !HCO3UseF = Value
                                  !SP 10/15/2007   Case(19)
                                        !BotAlgLightModel = Value
                                  Case(18)
                                        PAR   = Value
                                  Case(19)
                                        KLb = Value
                                  Case(20)
                                        khnxb = Value
                                        khnxb = khnxb*1.0e-03
      !convert from ugN/L to mgN/L
                                  Case(21)
                                        q0N = Value
                                        q0N = q0N*mgA/gD
      !convert from mgN/mgA to mgN/gD
                                  Case(22)
                                        q0P = Value
                                        q0P = q0P*mgA/gD
      !convert from mgP/mgA to mgP/gD
                                  Case(23)
                                        rmN =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                        rmN = rmN*mgA/gD
      !convert from mgN/mgA to mgN/gD
                                  Case(24)
                                        rmP =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                        rmP = rmP*mgA/gD
      !convert from mgP/mgA to mgP/gD
                                  Case(25)
                                        kqN = Value
                                        !SP 04/24/2008 Conversion moved
to main code
```

```
                                        !KqN = KqN*q0N
      !convert from unitless to mgN/gD

                                    Case(26)
                                        kqP = Value
                                        !SP 04/24/2008 Conversion moved
to main code
                                        !KqP = KqP*q0P
      !convert from unitless to mgP/gD

                                    Case(27)
                                        NUpWCFactor = Value
                                    Case(28)
                                        PUpWCFactor = Value
                                    !SP 10/15/2007   Case(29)
                                        !SP 10/15/2007   IkoxB  = Value
                                    Case(29)
                                        Ksob     = Value
                                    !SP 10/15/2007   Case(34)
                                        !Modeling_Method = Value
                                    !SP 10/15/2007   Case(35)
                                        !pCO2SetMethod = Value
                                    Case(30)
                                        pCO2 = Value
                                    !SP 10/15/2007   Case(37)
                                        !pHMethod = Value
                                    Case(31)
                                        K_LDOM =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(32)
                                        Tht_K_LDOM = Value
                                    Case(33)
                                        K_L2RDOMTR =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(34)
                                        K_RDOM     =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(35)
                                        Tht_K_RDOM = Value
                                    Case(36)
                                        K_LPOM =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(37)
                                        Tht_K_LPOM = Value
                                    Case(38)
                                        K_L2RPOMTR  =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(39)
                                        vs_LPOM =
VelocityConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(40)
                                        kds_LPOM  =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                    Case(41)
```

```
                                    Tht_kds_LPOM = Value
                                Case(42)
                                    Pam = Value
                                Case(43)
                                    K_RPOM =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                Case(44)
                                    Tht_K_RPOM  = Value
                                Case(45)
                                    vs_RPOM =
VelocityConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                Case(46)
                                    kds_RPOM =
RateConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                Case(47)
                                    Tht_kds_RPOM = Value
                                Case(48)
                                    r_OMN = Value
                                Case(49)
                                    r_OMP = Value
                                Case(50)
                                    r_OMC = Value
                                !SP 10/15/2007   Case(51)
                                    !SP 10/15/2007   TSSVelType =
Value
                                Case(51)
                                    TSSVel =
VelocityConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                Case(52)
                                    TSSSize =
DiameterConversion(Value,WQADD3D_iTVRCUnit(jdd,nci))
                                Case(53)
                                    TSSDensity = Value
                            End Select
                        End Do
                        Exit
                    Else
                        !
                        !vsk 01/04/00
                        If(WQADD3D_TVRCSTime(jd) > MdlTime) Return
                        Read(NFWQADD3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                    (WQADD3D_TVRCvalue(1,nci), nci = 1,
WQADD3D_nTVRCclms(jd))

                        Call
seterrormessage(istat,NFWQADD3D(jd),WQADD3D_TVRCFileName(jd))
                        If(istat < 0) WQADD3D_EndFileCountKDG(jd) =
WQADD3D_EndFileCountKDG(jd) + 1
                        If(istat > 0) then
                            GLLVHTError = 1
                            Return
                        End If
```

```
                        WQADD3D_TVRCSTime(jd) =
dfloat(dttm2julian(tvdsyear, tvdsmonth, tvdsday, tvdshour, tvdsmin))

                        Read(NFWQADD3D(jd),*,iostat=istat) tvdsyear,
tvdsmonth, tvdsday, tvdshour, tvdsmin,&
                  (WQADD3D_TVRCvalue(2,nci), nci = 1,
WQADD3D_nTVRCclms(jd))
                        Backspace(NFWQADD3D(jd))
                        Call
seterrormessage(istat,NFWQADD3D(jd),WQADD3D_TVRCFileName(jd))
                        If(istat < 0) WQADD3D_EndFileCountKDG(jd) =
WQADD3D_EndFileCountKDG(jd) + 1
                        If(istat > 0) Then
                              GLLVHTError = 1
                              Return
                        End If
                        WQADD3D_TVRCETime(jd) =
dfloat(dttm2julian(tvdsyear, tvdsmonth, tvdsday, tvdshour, tvdsmin))
                   End If
              End Do  !jd



      Return
      !
      !Additional water quality constituents
      ENTRY ReadWQADD3DControlFile


          NFwqctl = NFctl
        VariableCount = 0
           TVRCFileCount = 0
           Read (NFwqctl,'(//)')
        Read (NFwqctl,*,end=1000) message1, message2, j, j, j, i
           Allocate (WQADD3DA_cnss(ncWQADD,i+1), Stat = AllocError)
           i = 0
        Do While(.true.)
            Read (NFwqctl,*,end=1000) message
            Select Case(message(1:len_trim(message)-1))
                !
                !Rates and constants for Bottom Algae
                       Case('mgC')
                             BackSpace(NFwqctl)
                             Read(NFwqctl,*) message1, message2,
WQADD3D_mgC
                             Read(NFwqctl,*) message1, message2,
WQADD3D_mgN
                             Read(NFwqctl,*) message1, message2,
WQADD3D_mgP
                             Read(NFwqctl,*) message1, message2,
WQADD3D_mgD
                             Read(NFwqctl,*) message1, message2,
WQADD3D_mgA
```

```
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_MPHYT-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1

                              Read(NFwqctl,*) message1, message2,
WQADD3D_BotAlgGrowthModelType
                              !
                              !1/day to 1/sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Cgb20, Units
                              WQADD3D_Cgb20 =
RateConversion(WQADD3D_Cgb20, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_ThtCgb20
                              !
                              !mgA/m^2
                              Read(NFwqctl,*) message1, message2,
WQADD3D_abMax
                              !
                              !1/day to 1/sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_krb, Units
                              WQADD3D_krb = RateConversion(WQADD3D_krb,
Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Thtkrb
                              !
                              !1/day to 1/sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_keb, Units
                              WQADD3D_keb = RateConversion(WQADD3D_keb,
Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Thtkeb
                              !
                              !1/day to 1/sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_kdb, Units
                              WQADD3D_kdb = RateConversion(WQADD3D_kdb,
Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Thtkdb
                              !
                              !ug/l
                              Read(NFwqctl,*) message1, message2,
WQADD3D_ksNb
                              Read(NFwqctl,*) message1, message2,
WQADD3D_ksPb
                              Read(NFwqctl,*) message1, message2,
WQADD3D_ksCb

                              Read(NFwqctl,*) message1, message2,
WQADD3D_HCO3UseF
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3D_BotAlgLightModel
                              Read(NFwqctl,*) message1, message2,
WQADD3D_PAR
                              !
                              !leave it as per day
                              Read(NFwqctl,*) message1, message2,
WQADD3D_KLb

                              Read(NFwqctl,*) message1, message2,
WQADD3D_khnxb
                              !mgN/mgA
                              Read(NFwqctl,*) message1, message2,
WQADD3D_q0N
                              !mgP/mgA
                              Read(NFwqctl,*) message1, message2,
WQADD3D_q0P
                              !
                              !mgN/mgA-day to mgN/mgA-sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_rmN, Units
                              WQADD3D_rmN = RateConversion(WQADD3D_rmN,
Units)
                              !
                              !mgP/mgA-day to mgP/mgA-sec
                              Read(NFwqctl,*) message1, message2,
WQADD3D_rmP, Units
                              WQADD3D_rmP = RateConversion(WQADD3D_rmP,
Units)
                              !
                              !No units
                              Read(NFwqctl,*) message1, message2,
WQADD3D_KqN
                              Read(NFwqctl,*) message1, message2,
WQADD3D_KqP
                              Read(NFwqctl,*) message1, message2,
WQADD3D_NUpWCFactor
                              Read(NFwqctl,*) message1, message2,
WQADD3D_PUpWCFactor
                              !
                              !Switch
                              Read(NFwqctl,*) message1, message2,
WQADD3D_IkoxB
                              !
                              !l/mg O2
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Ksob

                              Read(NFwqctl,*) message1, message2,
WQADD3D_BotAlgModelingMethod
                              !
                              !pH
                              Read(NFwqctl,*) message1
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_pH-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_pHSolveMethod
                              !
                              !ALKL
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_ALKL-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2
                              !
                              !TIOC
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_TIOC-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_pCO2SetMethod
                              Read(NFwqctl,*) message1, message2,
WQADD3D_pCO2
                              !
                              !Convert from ppm to atm
                              WQADD3D_pCO2 = WQADD3D_pCO2/1.0d+06
                              !
                              !COND
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_COND-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2

                              !
                              !LDOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_LDOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_LDOM, Units
                              WQADD3D_K_LDOM =
RateConversion(WQADD3D_K_LDOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_LDOM

                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_L2RDOMTR, Units
                              WQADD3D_K_L2RDOMTR =
RateConversion(WQADD3D_K_L2RDOMTR, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_L2RDOMTR
                              !
                              !RDOM
```

```
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_RDOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_RDOM, Units
                              WQADD3D_K_RDOM =
RateConversion(WQADD3D_K_RDOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_RDOM
                              !
                              !LPOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_LPOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_LPOM, Units
                              WQADD3D_K_LPOM =
RateConversion(WQADD3D_K_LPOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_LPOM

                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_L2RPOMTR, Units
                              WQADD3D_K_L2RPOMTR =
RateConversion(WQADD3D_K_L2RPOMTR, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_L2RPOMTR

                              Read(NFwqctl,*) message1, message2,
WQADD3D_vs_LPOM, Units
                              WQADD3D_vs_LPOM =
VelocityConversion(WQADD3D_vs_LPOM, Units)

                              Read(NFwqctl,*) message1, message2,
WQADD3D_kds_LPOM, Units
                              WQADD3D_kds_LPOM =
RateConversion(WQADD3D_kds_LPOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_kds_LPOM

                              Read(NFwqctl,*) message1, message2,
WQADD3D_Pam
                              !
                              !RPOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_RPOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_K_RPOM, Units
                              WQADD3D_K_RPOM =
RateConversion(WQADD3D_K_RPOM, Units)
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_K_RPOM
                              Read(NFwqctl,*) message1, message2,
WQADD3D_vs_RPOM, Units
                              WQADD3D_vs_RPOM =
VelocityConversion(WQADD3D_vs_RPOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_kds_RPOM, Units
                              WQADD3D_kds_RPOM =
RateConversion(WQADD3D_kds_RPOM, Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_Tht_kds_RPOM

                              Read(NFwqctl,'(//)')
                              Read(NFwqctl,*) message1, message2,
WQADD3D_r_OMN
                              Read(NFwqctl,*) message1, message2,
WQADD3D_r_OMP
                              Read(NFwqctl,*) message1, message2,
WQADD3D_r_OMC
                              !
                              !TDS
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_TDS-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2
                              !
                              !TSS velocity type
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_SSS-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                        Read(NFwqctl,*) message1, message2,
WQADD3D_TSSVelType
                              !
                              !TSS velocity
                        Read(NFwqctl,*) message1, message2, WQADD3D_TSSVel,
Units
                              WQADD3D_TSSVel =
velocityconversion(WQADD3D_TSSVel, Units)
                              !
                              !TSS particle size
                        Read(NFwqctl,*) message1, message2, WQADD3D_TSSSize,
Units
                              WQADD3D_TSSSize =
DiameterConversion(WQADD3D_TSSSize,Units)
                              !
                              !TSS density
                        Read(NFwqctl,*) message1, message2,
WQADD3D_TSSDensity
                              !
                              !VSK 05-05-2005
                              Read(NFctl,'(//)')
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3D_UseTVRCFileLocal
                              Read(NFwqctl,*) message1, message2,
WQADD3D_TVRCFileNameLocal
                        Exit
               End Select

          End do

              Read(NFctl,*) message1, NumWQADD3DRegions

              TVRCFileCount = TVRCFileCount + 1
              WQADD3D_NumTVRCFiles = NumWQADD3DRegions + 1

              If(WQADD3D_NumTVRCFiles /= 0) Then
                    Allocate(WQADD3D_UseTVRCFile(WQADD3D_NumTVRCFiles), Stat
= AllocError)
                    Allocate(WQADD3D_TVRCFileName(WQADD3D_NumTVRCFiles),
Stat = AllocError)
                    Allocate(NFWQADD3D(WQADD3D_NumTVRCFiles), Stat =
AllocError)
              End If

              WQADD3D_TVRCFileName(1) = WQADD3D_TVRCFileNameLocal
              WQADD3D_UseTVRCFile(1) = WQADD3D_UseTVRCFileLocal

              If(NumWQADD3DRegions > 0) Then
                    Allocate(WQADD3DIStart(NumWQADD3DRegions),
WQADD3DJStart(NumWQADD3DRegions), Stat = AllocError)
                    Allocate(WQADD3DIEnd(NumWQADD3DRegions),
WQADD3DJEnd(NumWQADD3DRegions), Stat = AllocError)
                    Allocate(WQADD3DRegionStatus(im+1,jm+1), Stat =
AllocError)
                    Allocate(WQADD3DRegionRCDStatus(NumWQADD3DRegions), Stat
= AllocError)
                    Allocate(WQADD3DRegionName(NumWQADD3DRegions), Stat =
AllocError)
                    WQADD3DIStart = 0
                    WQADD3DJStart = 0
                    WQADD3DIEnd = 0
                    WQADD3DJEnd = 0
                    WQADD3DRegionStatus = 0
                    WQADD3DRegionRCDStatus = 0
                    Call AllocateWQADD3DWaterQualityVariables
              End If

              ChangesInRegions = .False.
              Do k = 1, NumWQADD3DRegions
                    Read(NFctl,*)
WQADD3DRegionName(k),WQADD3DIStart(k),WQADD3DIEnd(k),WQADD3DJStart(k),WQA
DD3DJEnd(k),i, j,WQADD3DRegionRCDStatus(k)
                    Do i = WQADD3DIStart(k)+1,WQADD3DIEnd(k)+1
                          Do j = WQADD3DJStart(k)+1,WQADD3DJEnd(k)+1
                                WQADD3DRegionStatus(i,j) = k
```

```
                  End Do
            End Do
            If(WQADD3DRegionRCDStatus(k) == 1) Then
                  ChangesInRegions = .True.
            End If

      End Do
      If(.Not. ChangesInRegions) NumWQADD3DRegions = 0

      Do i = 1, NumWQADD3DRegions
            If(WQADD3DRegionRCDStatus(i) == 0) Cycle
            Read(NFwqctl,*) message1, j, j, j, j
            Do While(.true.)
              Read (NFwqctl,*,end=1000) message
                Select Case(message(1:len_trim(message)-1))
                  Case('mgC')
                        BackSpace(NFwqctl)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_mgC(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_mgN(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_mgP(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_mgD(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_mgA(i)
                        !
                        !MPHYT
                        Read(NFwqctl,*) message1
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_MPHYT-ncWQADDSt+1,i+1)
                        Read(NFwqctl,*) message1
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_BotAlgGrowthModelType(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_Cgb20(i), Units
                        WQADD3DA_Cgb20(i) =
RateConversion(WQADD3DA_Cgb20(i), Units)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_ThtCgb20(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_abMax(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_krb(i), Units
                        WQADD3DA_krb(i) =
RateConversion(WQADD3DA_krb(i), Units)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_Thtkrb(i)
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_keb(i), Units
                        WQADD3DA_keb(i) =
RateConversion(WQADD3DA_keb(i), Units)
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Thtkeb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_kdb(i), Units
                              WQADD3DA_kdb(i) =
RateConversion(WQADD3DA_kdb(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Thtkdb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_ksNb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_ksPb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_ksCb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_HCO3UseF
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_BotAlgLightModel(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_PAR(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_KLb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_khnxb(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_q0N(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_q0P(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_rmN(i), Units
                              WQADD3DA_rmN(i) =
RateConversion(WQADD3DA_rmN(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_rmP(i), Units
                              WQADD3DA_rmP(i) =
RateConversion(WQADD3DA_rmP(i), Units)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_KqN(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_KqP(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_NUpWCFactor(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_PUpWCFactor(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_IkoxB(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Ksob(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_BotAlgModelingMethod(i)
                              !
```

```
                              !pH
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_pH-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_pHSolveMethod(i)
                              !
                              !ALKL
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_ALKL-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2
                              !
                              !TIOC
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_TIOC-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_pCO2SetMethod(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_pCO2(i)
                              !
                              !Convert from ppm to atm
                              WQADD3DA_pCO2(i) = WQADD3DA_pCO2(i)/1.0d+06

                              !
                              !COND
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_COND-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2
                              !
                              !LDOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_LDOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_LDOM(i), Units
                              WQADD3DA_K_LDOM(i) =
RateConversion(WQADD3DA_K_LDOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_K_LDOM(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_L2RDOMTR(i), Units
                              WQADD3DA_K_L2RDOMTR(i) =
RateConversion(WQADD3DA_K_L2RDOMTR(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_L2RDOMTR(i)
```

```
                              !
                              !RDOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_RDOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_RDOM(i), Units
                              WQADD3DA_K_RDOM(i) =
RateConversion(WQADD3DA_K_RDOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_K_RDOM(i)
                              !
                              !LPOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_LPOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_LPOM(i), Units
                              WQADD3DA_K_LPOM(i) =
RateConversion(WQADD3DA_K_LPOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_K_LPOM(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_L2RPOMTR(i), Units
                              WQADD3DA_K_L2RPOMTR(i) =
RateConversion(WQADD3DA_K_L2RPOMTR(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_K_L2RPOMTR(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_vs_LPOM(i), Units
                              WQADD3DA_vs_LPOM(i) =
VelocityConversion(WQADD3DA_vs_LPOM(i), Units)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_kds_LPOM(i), Units
                              WQADD3DA_kds_LPOM(i) =
RateConversion(WQADD3DA_kds_LPOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_kds_LPOM(i)

                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Pam(i)
                              !
                              !RPOM
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_RPOM-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
```

```
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_K_RPOM(i), Units
                              WQADD3DA_K_RPOM(i) =
RateConversion(WQADD3DA_K_RPOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_K_RPOM(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_vs_RPOM(i), Units
                              WQADD3DA_vs_RPOM(i) =
VelocityConversion(WQADD3DA_vs_RPOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_kds_RPOM(i), Units
                              WQADD3DA_kds_RPOM(i) =
RateConversion(WQADD3DA_kds_RPOM(i), Units)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_Tht_kds_RPOM(i)
                              Read(NFwqctl,'(//)')
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_r_OMN(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_r_OMP(i)
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_r_OMC(i)
                              !
                              !TDS
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_TDS-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2
                              !
                              !
                              !TSS velocity type
                              Read(NFwqctl,*) message1
                              Read(NFwqctl,*) message1, message2,
WQADD3DA_cnss(I_SSS-ncWQADDSt+1,i+1)
                              Read(NFwqctl,*) message1
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_TSSVelType(i)
                              !
                              !TSS velocity
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_TSSVel(i), Units
                              WQADD3DA_TSSVel =
velocityconversion(WQADD3DA_TSSVel(i), Units)
                              !
                              !TSS particle size
                        Read(NFwqctl,*) message1, message2,
WQADD3DA_TSSSize(i), Units
                              WQADD3DA_TSSSize =
DiameterConversion(WQADD3DA_TSSSize(i),Units)
                              !
                              !TSS density
```

```
                    Read(NFwqctl,*) message1, message2,
WQADD3DA_TSSDensity(i)
                              Read(NFctl,'(//)')
                              TVRCFileCount = TVRCFileCount + 1
                              Read(NFwqctl,*) message1, message2,
WQADD3D_UseTVRCFile(TVRCFileCount)
                              Read(NFwqctl,*) message1, message2,
WQADD3D_TVRCFileName(TVRCFileCount)
                    Exit
                       End Select
                 End Do
        End do
            !
            !
            If(WQADD3D_NumTVRCFiles /= 0) Call OpenWQADD3DTVRCFiles
            Return
1000  message = 'Error in Reading Rates and Constants for WQADD, Please
check the control file'C
            iretw = MessageBox(GetActiveWindow(),message,'GEMSS
Model'C,MB_ICONINFORMATION.or.MB_OK)
            Stop
    Return

      Entry AllocateWQADD3DWaterQualityVariables
            !
            !Macrophytes
            Allocate (WQADD3DA_mgC(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate (WQADD3DA_mgN(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate (WQADD3DA_mgP(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate (WQADD3DA_mgD(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate (WQADD3DA_mgA(NumWQADD3DRegions+1),       Stat =
AllocError)

            Allocate(WQADD3DA_Cgb20(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate(WQADD3DA_ThtCgb20(NumWQADD3DRegions+1),    Stat =
AllocError)
            Allocate(WQADD3DA_abMax(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate(WQADD3DA_krb(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_Thtkrb(NumWQADD3DRegions+1),      Stat =
AllocError)
            Allocate(WQADD3DA_keb(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_Thtkeb(NumWQADD3DRegions+1),      Stat =
AllocError)
            Allocate(WQADD3DA_kdb(NumWQADD3DRegions+1),         Stat =
AllocError)
```

```
            Allocate(WQADD3DA_Thtkdb(NumWQADD3DRegions+1),      Stat =
AllocError)
            Allocate(WQADD3DA_ksPb(NumWQADD3DRegions+1),        Stat =
AllocError)
            Allocate(WQADD3DA_ksNb(NumWQADD3DRegions+1),        Stat =
AllocError)
            Allocate(WQADD3DA_ksCb(NumWQADD3DRegions+1),        Stat =
AllocError)
            Allocate(WQADD3DA_NUpWCFactor(NumWQADD3DRegions+1), Stat =
AllocError)
            Allocate(WQADD3DA_PUpWCFactor(NumWQADD3DRegions+1), Stat =
AllocError)
            Allocate(WQADD3DA_Ksob(NumWQADD3DRegions+1),        Stat =
AllocError)
            Allocate(WQADD3DA_IkoxB(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate(WQADD3DA_khnxb(NumWQADD3DRegions+1),       Stat =
AllocError)
            Allocate(WQADD3DA_KLb(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_q0N(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_q0P(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_rmN(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_rmP(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_KqN(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_KqP(NumWQADD3DRegions+1),         Stat =
AllocError)
            Allocate(WQADD3DA_PAR(NumWQADD3DRegions+1),              Stat
= AllocError)

            Allocate(WQADD3DA_BotAlgLightModel(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_BotAlgModelingMethod(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_BotAlgGrowthModelType(NumWQADD3DRegions+1),
      Stat = AllocError)

            Allocate(WQADD3DA_K_LDOM(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_Tht_K_LDOM(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_K_L2RDOMTR(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_K_RDOM(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_Tht_K_RDOM(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_K_LPOM(NumWQADD3DRegions+1),
      Stat = AllocError)
```

```
            Allocate(WQADD3DA_Tht_K_LPOM(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_K_L2RPOMTR(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_vs_LPOM(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_kds_LPOM(NumWQADD3DRegions+1),         Stat
= AllocError)
            Allocate(WQADD3DA_kds_LPOM(NumWQADD3DRegions+1),         Stat
= AllocError)
            Allocate(WQADD3DA_Tht_kds_LPOM(NumWQADD3DRegions+1),     Stat
= AllocError)
            Allocate(WQADD3DA_Pam(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_K_RPOM(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_Tht_K_RPOM(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_vs_RPOM(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_kds_RPOM(NumWQADD3DRegions+1),         Stat
= AllocError)
            Allocate(WQADD3DA_Tht_kds_RPOM(NumWQADD3DRegions+1),     Stat
= AllocError)
            Allocate(WQADD3DA_TSSVelType(NumWQADD3DRegions+1),       Stat
= AllocError)
            Allocate(WQADD3DA_TSSVel(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_TSSSize(NumWQADD3DRegions+1),
      Stat = AllocError)
            Allocate(WQADD3DA_TSSDensity(NumWQADD3DRegions+1),       Stat
= AllocError)


            Allocate(WQADD3DA_pHSolveMethod(NumWQADD3DRegions+1),    Stat
= AllocError)
            Allocate(WQADD3DA_r_OMN(NumWQADD3DRegions+1),            Stat
= AllocError)
            Allocate(WQADD3DA_r_OMP(NumWQADD3DRegions+1),            Stat
= AllocError)
            Allocate(WQADD3DA_r_OMC(NumWQADD3DRegions+1),            Stat
= AllocError)

            Allocate(WQADD3DA_Tht_K_L2RPOMTR(NumWQADD3DRegions+1),   Stat
= AllocError)
            Allocate(WQADD3DA_Tht_K_L2RDOMTR(NumWQADD3DRegions+1),   Stat
= AllocError)

            Allocate(WQADD3DA_pCO2SetMethod(NumWQADD3DRegions+1), Stat =
AllocError)
            Allocate(WQADD3DA_pCO2(NumWQADD3DRegions+1), Stat =
AllocError)

            Allocate(MGM3D_UDC_Output(2,13))
```

```
            MGM3D_UDC_Output = 0.d00

            Allocate(WQADD_3D_nt(n1+1), WQADD_3D_ps(n1+1),
WQADD_3D_Lb(n1+1), WQADD_3D_Sb(n1+1), WQADD_3D_nb(n1+1))
            WQADD_3D_nt = 0.d00
            WQADD_3D_ps = 0.d00
            WQADD_3D_Lb = 0.d00
            WQADD_3D_Sb = 0.d00
            WQADD_3D_nb = 0.d00



      Return

    Entry WriteWQADD3DControlFileToSnapShot
            !
            !Write Parameters for the whole waterbody
        Write(NFSnp,'(/)')
            Write(NFSnp,'(a)') &

      '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@'
            Write(NFSnp,'(a)') &
            '                               GEMSS-WQADD
'
            Write(NFSnp,'(a)') &
            '     Additional Water Quality Model Kinetics Rates and
Constants       '
        Write(NFSnp,'(a)') &

      '@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@'
        Write(NFSnp,'(/)')

            Write(NFSnp,'(a)') &

      '*****************************************************************
******'
            Write(NFSnp,'(a)') 'Region: Whole Waterbody'
        Write(NFSnp,'(a)') &

      '*****************************************************************
******'
        Write(NFSnp,'(/)')

          Write(NFSnp,'(a)') &

'################################################################################
'
        Write(NFSnp,'(a)') &
        'I_MPHYT'
        Write(NFSnp,'(a)') &
```

```
'###########################################################################
'
        Write(NFSnp,'(a,e13.6,a)') &
        'mgC,   Carbon stoichiometric value
=', WQADD3D_mgC,  ' gC'
        Write(NFSnp,'(a,e13.6,a)') &
        'mgN,   Nitrogen stoichiometric value
=', WQADD3D_mgN,  ' gN'
        Write(NFSnp,'(a,e13.6,a)') &
        'mgP,   Phosphorous stoichiometric value
=', WQADD3D_mgP,  ' gP'
        Write(NFSnp,'(a,e13.6,a)') &
        'mgD,   Dry weight
=', WQADD3D_mgD,  ' gD'
        Write(NFSnp,'(a,e13.6,a)') &
        'mgA,   Temperature coefficient
=', WQADD3D_mgA,  ' gA'

           If(WQADD3D_BotAlgGrowthModelType == 1) Then
           Write(NFSnp,'(a,a,a)') &
           'Bottom Alage growth model type
=', ' Zero Order'
           Else
           Write(NFSnp,'(a,a,a)') &
           'Bottom Alage growth model type
=', ' First Order'
           End If

           If(WQADD3D_BotAlgGrowthModelType == 1) Then
           Write(NFSnp,'(a,e13.6,a)') &
         'Cgb20, Maximum growth rate
=', WQADD3D_Cgb20, ' mgA/m^2-sec'
        Else
          Write(NFSnp,'(a,e13.6,a)') &
            'Cgb20, Maximum growth rate
=', WQADD3D_Cgb20, ' 1/sec'
           End If

           Write(NFSnp,'(a,e13.6,a)') &
        'ThtCgb20, Temperature correction for growth rate
=', WQADD3D_ThtCgb20, ' '
           Write(NFSnp,'(a,e13.6,a)') &
        'abMax, Temperature correction for growth rate
=', WQADD3D_abMax, ' mgA/m^2'
           Write(NFSnp,'(a,e13.6,a)') &
        'krb, Respiration rate
=', WQADD3D_krb, ' 1/sec'
           Write(NFSnp,'(a,e13.6,a)') &
        'Thtkrb, Temperature correction for respiration
=', WQADD3D_Thtkrb, ' '
           Write(NFSnp,'(a,e13.6,a)') &
        'keb, Temperature correction for excretion
=', WQADD3D_keb, ' 1/sec'
```

```
            Write(NFSnp,'(a,e13.6,a)') &
          'Thtkeb, Temperature correction for respiration
=', WQADD3D_Thtkeb, ' '
            Write(NFSnp,'(a,e13.6,a)') &
          'kdb, Temperature correction for excretion
=', WQADD3D_kdb, ' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
          'Thtkdb, Temperature correction for respiration
=', WQADD3D_Thtkdb, ' '

            Write(NFSnp,'(a,e13.6,a)') &
          'ksNb, External nitrogen half saturation constant
=', WQADD3D_ksNb, ' gN/m^3'
            Write(NFSnp,'(a,e13.6,a)') &
          'ksPb, External nitrogen half saturation constant
=', WQADD3D_ksPb, ' gP/m^3'
            Write(NFSnp,'(a,e13.6,a)') &
           'ksCb, Inorganic carbon half saturation constant
=', WQADD3D_ksCb, ' gC/m^3'
            !
            !Light model
            Select Case(WQADD3D_BotAlgLightModel)
                  Case(1)
                        Write(NFSnp,'(a,a,a)') &
            'BotAlgLightModel, Light model
=', ' Half-Saturation Light Mode', ' '
                  Case(2)
                        Write(NFSnp,'(a,a,a)') &
            'BotAlgLightModel, Light model
=', " Smith's Function (Smith 1936)", ' '
                  Case(3)
                        Write(NFSnp,'(a,a,a)') &
            'BotAlgLightModel, Light model
=', " Steele's Equation (Steele 1962)", ' '
            End Select

            If(WQADD3D_HCO3useF == 0) Then
                Write(NFSnp,'(a,a,a)') &
            'Use HCO3 subtrate in bottom algea kinetics
=', ' No', ' '
            Else
                Write(NFSnp,'(a,a,a)') &
            'Use HCO3 subtrate in bottom algea kinetics
=', ' Yes', ' '
            End If

            Write(NFSnp,'(a,e13.6,a)') &
            'PAR, Photosynthetically available radiation
=', WQADD3D_PAR,''
            Write(NFSnp,'(a,e13.6,a)') &
            'KLb, Light constant
=', WQADD3D_KLb,' Langleys/day'
            Write(NFSnp,'(a,e13.6,a)') &
```

```
          'khnxb, Ammonia preference
=', WQADD3D_khnxb, ' ugN/L'
            Write(NFSnp,'(a,e13.6,a)') &
          'q0N, Subsistence quota for nitrogen
=', WQADD3D_q0N, ' mgN/mgA'
            Write(NFSnp,'(a,e13.6,a)') &
          'q0P, Subssitence quota for phosphorous
=', WQADD3D_q0P, ' mgP/mgA'
            Write(NFSnp,'(a,e13.6,a)') &
          'rmN, Maximum uptake rate for nitrogen
=', WQADD3D_rmN, ' mgN/mgA-Sec'
            Write(NFSnp,'(a,e13.6,a)') &
          'rmP, Maximum uptake rate for nitrogen
=', WQADD3D_rmP, ' mgP/mgA-Sec'
            Write(NFSnp,'(a,e13.6,a)') &
          'KqN, Internal nitrogen half saturation ratio
=', WQADD3D_KqN, ' mgN/mgA-Sec'
            Write(NFSnp,'(a,e13.6,a)') &
          'KqP, Internal phosphorous half saturation ratio
=', WQADD3D_KqP, ' mgP/mgA-Sec'


            Write(NFSnp,'(a,e13.6,a)') &
          'NUpWCFactor, Nitrogen uptake water column factor
=', WQADD3D_NUpWCFactor, ' '
            Write(NFSnp,'(a,e13.6,a)') &
          'PUpWCFactor, Phosphorous uptake water column factor
=', WQADD3D_PUpWCFactor, ' '

            Select Case(WQADD3D_Ikoxb)
                  Case(1)
                        Write(NFSnp,'(a,a,a)') &
            'Oxygen enhanced model for bottom algae respiration
=', ' Exponential', ' '
                  Case(2)
                        Write(NFSnp,'(a,a,a)') &
            'Oxygen enhanced model for bottom algae respiration
=', ' Half-Saturation', ' '
                  Case(3)
                        Write(NFSnp,'(a,a,a)') &
            'Oxygen enhanced model for bottom algae respiration
=', ' Second Order', ' '
            End Select

            Write(NFSnp,'(a,e13.6,a)') &
            'Ksob, Oxygen enhance parameter for bottom algae respiration
=', WQADD3D_Ksob, ' '

            If(WQADD3D_BotAlgModelingMethod == 1) Then
                  Write(NFSnp,'(a,a,a)') &
            'BotAlgModelingMethod, Bottom algae modeling method
=', ' QUAL2Kw', '  '
            Else
                  Write(NFSnp,'(a,a,a)') &
```

```
            'BotAlgModelingMethod, Bottom algae modeling method
=', ' CE-QUAL-W2', '  '
            End If

            Write(NFSnp,'(a)') &

      '#################################################################
#########################################'
            Write(NFSnp,'(a)') &
            'I_pH, pH and associated variables: I_ALKL; I_TIOC; I_COND;
Alkalinity; Total inorganic carbon and conductivity'
            Write(NFSnp,'(a)') &

      '#################################################################
#########################################'
            If(pCO2SetMethod == 1) Then
            Write(NFSnp,'(a,a,a)') &
            'Method to specify partial pressure of carbon dioxide
=',' Compute',' '
            Else
            Write(NFSnp,'(a,a,a)') &
            'Method for partial pressure of carbon dioxide
=',' User Defined',' '
            End If
            Write(NFSnp,'(a,e13.6,a)') &
            'pCO2, Partial pressure of carbon dioxide
=', WQADD3D_pCO2, ' ppm'
            If(pHSolveMethod == 1) Then
            Write(NFSnp,'(a,a,a)') &
            'Method of pH computation
=', ' Newton-Rapshon', ' '
            Else If(pHSolveMethod == 2) Then
            Write(NFSnp,'(a,a,a)') &
            'Method of pH computation
=', ' Bisection', ' '
            End If
            Write(NFSnp,'(a)') &

      '#################################################################
####'
            Write(NFSnp,'(a)') &
            'I_LDOM, Labile Dissolved Organic Matter'
            Write(NFSnp,'(a)') &

      '#################################################################
####'
            Write(NFSnp,'(a,e13.6,a)') &
            'K_LDOM, Labile DOM decay rate
=', WQADD3D_K_LDOM, ' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_K_LDOM, Temperature correction factor for labile DOM
=', WQADD3D_Tht_K_LDOM, '  '
            Write(NFSnp,'(a,e13.6,a)') &
```

```
            'K_L2RDOMTR, Labile to refractory DOM transfer rate
=', WQADD3D_K_L2RDOMTR,' 1/sec'

            Write(NFSnp,'(a)') &

      '#####################################################################
####'
            Write(NFSnp,'(a)') &
            'I_RDOM, Refractory Dissolved Organic Matter'
            Write(NFSnp,'(a)') &

      '#####################################################################
####'
            Write(NFSnp,'(a,e13.6,a)') &
            'K_RDOM, Refractory DOM decay rate
=',WQADD3D_K_RDOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_K_RDOM, Temperature correction factor for refractory DOM
decay rate      =',WQADD3D_Tht_K_RDOM,'  '

            Write(NFSnp,'(a)') &

      '#####################################################################
####'
            Write(NFSnp,'(a)') &
            'I_LPOM, Labile particulate organic matter'
            Write(NFSnp,'(a)') &

      '#####################################################################
####'
            Write(NFSnp,'(a,e13.6,a)') &
            'K_LPOM, Labile POM decay rate
=',WQADD3D_K_LPOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_K_LPOM, Temperature correction factor for labile POM
decay rate          =',WQADD3D_Tht_K_LPOM,' '
            Write(NFSnp,'(a,e13.6,a)') &
            'K_L2RPOMTR, Labile to refractory POM transfer rate
=',WQADD3D_K_L2RPOMTR,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'vs_LPOM, Settling velocity of labile POM
=',WQADD3D_vs_LPOM, ' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'kds_LPOM, Dissolution rate of labile POM
=',WQADD3D_kds_LPOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_kds_LPOM, Temperature correction factor for labile POM
dissolution rate  =',WQADD3D_Tht_kds_LPOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Pam, Fraction of algal biomass that is converted into POM
=',WQADD3D_Pam,' '

            Write(NFSnp,'(a)') &
```

```
      '###################################################################
####'
            Write(NFSnp,'(a)') &
            'I_RPOM, Refractory particulate organic matter'
            Write(NFSnp,'(a)') &

      '###################################################################
####'
            Write(NFSnp,'(a,e13.6,a)') &
            'K_RPOM, Refractory DOM decay rate
=',WQADD3D_K_RPOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_K_RPOM, Temperature correction factor for refractory DOM
decay rate      =',WQADD3D_Tht_K_RPOM,'  '
            Write(NFSnp,'(a,e13.6,a)') &
            'vs_RPOM, Settling velocity of refractory POM
=',WQADD3D_vs_RPOM,' m/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'kds_RPOM, Dissolution rate of refractory POM
=',WQADD3D_kds_RPOM,' 1/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'Tht_kds_RPOM, Temp  correction factor for refractory POM
dissolution         =',WQADD3D_Tht_kds_RPOM

            Write(NFSnp,'(a)') &

      '###################################################################
####'
            Write(NFSnp,'(a)') &
            'Stoichiometry, Stoichiometry for organic matter'
            Write(NFSnp,'(a)') &

      '###################################################################
####'
            Write(NFSnp,'(a,e13.6,a)') &
            'r_OMN, Stoichiometric equivalent between organic matter and
nitrogen         =',WQADD3D_r_OMN,'  '
            Write(NFSnp,'(a,e13.6,a)') &
            'r_OMP, Stoichiometric equivalent between organic matter and
phosphorous      =',WQADD3D_r_OMP,'  '
            Write(NFSnp,'(a,e13.6,a)') &
            'r_OMC, Stoichiometric equivalent between organic matter and
carbon           =',WQADD3D_r_OMC,'  '

            Write(NFSnp,'(a)') &

      '###################################################################
####'
            Write(NFSnp,'(a)') &
            'TSS, Total suspended solids'
            Write(NFSnp,'(a)') &
```

```
      '#################################################################
####'
            If(WQADD3D_TSSVelType == 0) Then
            Write(NFSnp,'(a,a,a)') &
            'Total suspended solids velocity type
=',' User Supplied Velocity', ' '
            Else
            Write(NFSnp,'(a,a,a)') &
            'Total suspended solids velocity type
=',' Model Computation', '  '
            End If
            Write(NFSnp,'(a,e13.6,a)') &
            'TSSVel, Total suspended solids velocity
=',WQADD3D_TSSVel,' m/sec'
            Write(NFSnp,'(a,e13.6,a)') &
            'TSSSize, Total suspended solids size
=',WQADD3D_TSSSize, ' m'
            Write(NFSnp,'(a,e13.6,a)') &
            'TSSDensity, Total suspended solids density
=',WQADD3D_TSSDensity, ' kg/m^3'
            !
            !Write variables if regions are used
            If(NumWQADD3DRegions > 0) Then
                  Do i = 1, NumWQADD3DRegions
                      Write(NFSnp,'(/)')
                        Write(NFSnp,'(a)') &

      '*****************************************************************
******'
                        Write(NFSnp,'(a,a)') 'Region: ',
WQADD3DRegionName(i)
                      Write(NFSnp,'(a)') &

      '*****************************************************************
******'
                      Write(NFSnp,'(/)')
                        Write(NFSnp,'(a)') &

      '#################################################################
######'
                        Write(NFSnp,'(a)') &
                        'I_MPHYT'
                        Write(NFSnp,'(a)') &

      '#################################################################
######'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'mgC,   Carbon stoichiometric value
=', WQADD3DA_mgC(i),  ' gC'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'mgN,   Nitrogen stoichiometric value
=', WQADD3DA_mgN(i),  ' gN'
                        Write(NFSnp,'(a,e13.6,a)') &
```

```
                        'mgP,   Phosphorous stoichiometric value
=', WQADD3DA_mgP(i),  ' gP'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'mgD,   Dry weight
=', WQADD3DA_mgD(i),  ' gD'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'mgA,   Temperature coefficient
=', WQADD3DA_mgA(i),  ' gA'

                        If(WQADD3D_BotAlgGrowthModelType == 1) Then
                             Write(NFSnp,'(a,a,a)') &
                        'Bottom Alage growth model type
=', ' Zero Order'
                        Else
                             Write(NFSnp,'(a,a,a)') &
                        'Bottom Alage growth model type
=', ' First Order'
                        End If

                        If(WQADD3D_BotAlgGrowthModelType == 1) Then
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Cgb20, Maximum growth rate
=', WQADD3DA_Cgb20(i), ' mgA/m^2-sec'
                        Else
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Cgb20, Maximum growth rate
=', WQADD3DA_Cgb20(i), ' 1/sec'
                        End If

                        Write(NFSnp,'(a,e13.6,a)') &
                        'ThtCgb20, Temperature correction for growth rate
=', WQADD3DA_ThtCgb20(i), ' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'abMax, Temperature correction for growth rate
=', WQADD3DA_abMax(i), ' mgA/m^2'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'krb, Respiration rate
=', WQADD3DA_krb(i), ' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Thtkrb, Temperature correction for respiration
=', WQADD3DA_Thtkrb(i), ' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'keb, Temperature correction for excretion
=', WQADD3DA_keb(i), ' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Thtkeb, Temperature correction for respiration
=', WQADD3DA_Thtkeb(i), ' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'kdb, Temperature correction for excretion
=', WQADD3DA_kdb(i), ' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Thtkdb, Temperature correction for respiration
=', WQADD3DA_Thtkdb(i), ' '
```

```
                      Write(NFSnp,'(a,e13.6,a)') &
                      'ksNb, External nitrogen half saturation constant
=', WQADD3DA_ksNb(i), ' gN/m^3'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'ksPb, External nitrogen half saturation constant
=', WQADD3DA_ksPb(i), ' gP/m^3'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'ksCb, Inorganic carbon half saturation constant
=', WQADD3DA_ksCb(i), ' gC/m^3'
                      !
                      !Light model
                      Select Case(WQADD3D_BotAlgLightModel)
                            Case(1)
                                  Write(NFSnp,'(a,a,a)') &
                      'BotAlgLightModel, Light model
=', ' Half-Saturation Light Mode', ' '
                            Case(2)
                                  Write(NFSnp,'(a,a,a)') &
                      'BotAlgLightModel, Light model
=', " Smith's Function (Smith 1936)", ' '
                            Case(3)
                                  Write(NFSnp,'(a,a,a)') &
                      'BotAlgLightModel, Light model
=', " Steele's Equation (Steele 1962)", ' '
                      End Select

                      If(WQADD3D_HCO3useF == 0) Then
                            Write(NFSnp,'(a,a,a)') &
                      'Use HCO3 subtrate in bottom algea kinetics
=', ' No', ' '
                      Else
                            Write(NFSnp,'(a,a,a)') &
                      'Use HCO3 subtrate in bottom algea kinetics
=', ' Yes', ' '
                      End If

                      Write(NFSnp,'(a,e13.6,a)') &
                      'PAR, Photosynthetically available radiation
=', WQADD3DA_PAR(i),''
                      Write(NFSnp,'(a,e13.6,a)') &
                      'KLb, Light constant
=', WQADD3DA_KLb(i),' Langleys/day'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'khnxb, Ammonia preference
=', WQADD3DA_khnxb(i), ' ugN/L'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'q0N, Subsistence quota for nitrogen
=', WQADD3DA_q0N(i), ' gN/gA'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'q0P, Subssitence quota for phosphorous
=', WQADD3DA_q0P(i), ' gP/gA'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'rmN, Maximum uptake rate for nitrogen
=', WQADD3DA_rmN(i), ' gN/gA-Sec'
```

```
                        Write(NFSnp,'(a,e13.6,a)') &
                        'rmP, Maximum uptake rate for nitrogen
=', WQADD3DA_rmP(i), ' gP/gA-Sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'KqN, Internal nitrogen half saturation ratio
=', WQADD3DA_KqN(i), ' gN/gA-Sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'KqP, Internal phosphorous half saturation ratio
=', WQADD3DA_KqP(i), ' gP/gA-Sec'


                        Write(NFSnp,'(a,i13.6,a)') &
                        'NUpWCFactor, Nitrogen uptake water column factor
=', WQADD3DA_NUpWCFactor(i), ' '
                        Write(NFSnp,'(a,i13.6,a)') &
                        'PUpWCFactor, Phosphorous uptake water column
factor                          =', WQADD3DA_PUpWCFactor(i), ' '

                        Select Case(WQADD3D_Ikoxb)
                             Case(1)
                                  Write(NFSnp,'(a,a,a)') &
                        'Oxygen enhanced model for bottom algae
respiration                          =', ' Exponential', ' '
                             Case(2)
                                  Write(NFSnp,'(a,a,a)') &
                        'Oxygen enhanced model for bottom algae
respiration                          =', ' Half-Saturation', ' '
                             Case(3)
                                  Write(NFSnp,'(a,a,a)') &
                        'Oxygen enhanced model for bottom algae
respiration                          =', ' Second Order', ' '
                        End Select

                        Write(NFSnp,'(a,e13.6,a)') &
                        'Ksob, Oxygen enhance parameter for bottom algae
respiration                =', WQADD3DA_Ksob(i), ' '

                        If(WQADD3D_BotAlgModelingMethod == 1) Then
                             Write(NFSnp,'(a,a,a)') &
                        'BotAlgModelingMethod, Bottom algae modeling
method                           =', ' QUAL2Kw', '  '
                        Else
                             Write(NFSnp,'(a,a,a)') &
                        'BotAlgModelingMethod, Bottom algae modeling
method                           =', ' CE-QUAL-W2', '  '
                        End If

                        Write(NFSnp,'(a)') &

     '#########################################################################
#########################################'
                        Write(NFSnp,'(a)') &
                        'I_pH, pH and associated variables: I_ALKL;
I_TIOC; I_COND; Alkalinity; Total inorganic carbon and conductivity'
```

```
                      Write(NFSnp,'(a)') &

     '#################################################################
##########################################'
                      If(pCO2SetMethod == 1) Then
                           Write(NFSnp,'(a,a,a)') &
                      'Method to specifiy partial pressure of carbon
dioxide                      =', ' Compute', ' '
                      Else
                           Write(NFSnp,'(a,a,a)') &
                      'Method for partial pressure of carbon dioxide
=', ' User Defined', ' '
                      End If
                      Write(NFSnp,'(a)') &
                      'pCO2, Partial pressure of carbon dioxide
=', WQADD3DA_pCO2(i), ' ppm'
                      If(pHSolveMethod == 1) Then
                      Write(NFSnp,'(a)') &
                      'Method fo pH computation
=', ' Newton-Rapshon', ' '
                      Else If(pHSolveMethod == 2) Then
                      Write(NFSnp,'(a)') &
                      'Method fo pH computation
=', ' Bisection', ' '
                      End If
                      Write(NFSnp,'(a)') &

     '#################################################################
####'
                      Write(NFSnp,'(a)') &
                      'I_LDOM, Labile Dissolved Organic Matter'
                      Write(NFSnp,'(a)') &

     '#################################################################
####'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'K_LDOM, Labile DOM decay rate
=', WQADD3DA_K_LDOM(i), ' 1/sec'
                      Write(NFSnp,'(a,e13.6,a)') &
                      'Tht_K_LDOM, Temperature correction factor for
labile DOM                      =', WQADD3DA_Tht_K_LDOM(i), '  '
                      Write(NFSnp,'(a,e13.6,a)') &
                      'K_L2RDOMTR, Labile to refractory DOM transfer
rate                      =', WQADD3DA_K_L2RDOMTR(i),' 1/sec'

                      Write(NFSnp,'(a)') &

     '#################################################################
####'
                      Write(NFSnp,'(a)') &
                      'I_RDOM, Refractory Dissolved Organic Matter'
                      Write(NFSnp,'(a)') &
```

```
      '#####################################################################
####'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'K_RDOM, Refractory DOM decay rate
=',WQADD3DA_K_RDOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_K_RDOM, Temperature correction factor for
refractory DOM decay rate      =',WQADD3DA_Tht_K_RDOM(i),'  '

                        Write(NFSnp,'(a)') &

      '#####################################################################
####'
                        Write(NFSnp,'(a)') &
                        'I_LPOM, Labile particulate organic matter'
                        Write(NFSnp,'(a)') &

      '#####################################################################
####'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'K_LPOM, Labile POM decay rate
=',WQADD3DA_K_LPOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_K_LPOM, Temperature correction factor for
labile POM decay rate          =',WQADD3DA_Tht_K_LPOM(i),' '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'K_L2RPOMTR, Labile to refractory POM transfer
rate                           =',WQADD3DA_K_L2RPOMTR(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'vs_LPOM, Settling velocity of labile POM
=',WQADD3DA_vs_LPOM(i), ' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'kds_LPOM, Dissolution rate of labile POM
=',WQADD3DA_kds_LPOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_kds_LPOM, Temperature correction factor for
labile POM dissolution rate  =',WQADD3DA_Tht_kds_LPOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Pam, Fraction of algal biomass that is converted
into POM                   =',WQADD3DA_Pam(i),' '

                        Write(NFSnp,'(a)') &

      '#####################################################################
####'
                        Write(NFSnp,'(a)') &
                        'I_RPOM, Refractory particulate organic matter'
                        Write(NFSnp,'(a)') &

      '#####################################################################
####'
                        Write(NFSnp,'(a,e13.6,a)') &
```

```
                        'K_RPOM, Refractory DOM decay rate
=',WQADD3DA_K_RPOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_K_RPOM, Temperature correction factor for
refractory DOM decay rate      =',WQADD3DA_Tht_K_RPOM(i),'  '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'vs_RPOM, Settling velocity of refractory POM
=',WQADD3DA_vs_RPOM(i),' m/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'kds_RPOM, Dissolution rate of refractory POM
=',WQADD3DA_kds_RPOM(i),' 1/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'Tht_kds_RPOM, Temp  correction factor for
refractory POM dissolution         =',WQADD3DA_Tht_kds_RPOM(i)

                        Write(NFSnp,'(a)') &

      '#################################################################
####'
                        Write(NFSnp,'(a)') &
                        'Stoichiometry, Stoichiometry for organic matter'
                        Write(NFSnp,'(a)') &

      '#################################################################
####'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'r_OMN, Stoichiometric equivalent between organic
matter and nitrogen         =',WQADD3DA_r_OMN(i),'  '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'r_OMP, Stoichiometric equivalent between organic
matter and phosphorous      =',WQADD3DA_r_OMP(i),'  '
                        Write(NFSnp,'(a,e13.6,a)') &
                        'r_OMC, Stoichiometric equivalent between organic
matter and carbon           =',WQADD3DA_r_OMC(i),'  '

                        Write(NFSnp,'(a)') &

      '#################################################################
####'
                        Write(NFSnp,'(a)') &
                        'TSS, Total suspended solids'
                        Write(NFSnp,'(a)') &

      '#################################################################
####'
                        If(WQADD3D_TSSVelType == 0) Then
                        Write(NFSnp,'(a,a,a)') &
                        'Total suspended solids velocity type
=',' User Supplied Velocity', ' '
                        Else
                        Write(NFSnp,'(a,a,a)') &
                        'Total suspended solids velocity type
=',' Model Computation', '  '
                        End If
```

```
                        Write(NFSnp,'(a,e13.6,a)') &
                        'TSSVel, Total suspended solids velocity
='',WQADD3DA_TSSVel(i),' m/sec'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'TSSSize, Total suspended solids size
='',WQADD3DA_TSSSize(i), ' m'
                        Write(NFSnp,'(a,e13.6,a)') &
                        'TSSDensity, Total suspended solids density
='',WQADD3DA_TSSDensity(i), ' kg/m^3'
                  End Do
            End If
    Return

      Entry phco2FromW2
            !
            !ph and carbonate species
            cart = c(n,k,I_TIOC)/12000.0
            alkt = c(n,k,I_ALKL)/5.0e+04
            t1k  = c(n,k,I_Temp) + 273.15
            !
            !**** ionic strength
            If (.Not. Saline) s2 = 2.5e-05*c(n,k,I_TDS)
            If (Saline)  s2 = 1.47e-3 + 1.9885e-2*c(n,k,I_Saln) + 3.8e-
5*c(n,k,I_Saln)
            !
            !**** debye-huckel terms and activity coefficients
            sqrs2  =  sqrt(s2)
            dh1    = -0.5085*sqrs2/(1.0+1.3124*sqrs2)+4.745694e-
03+4.160762e-02*s2-9.284843e-03*s2*s2
            dh2    = -2.0340*sqrs2/(1.0+1.4765*sqrs2)+1.205665e-
02+9.715745e-02*s2-2.067746e-02*s2*s2
            h2co3t =  10.0**(0.0755*s2)
            hco3t  =  10.0**dh1
            co3t   =  10.0**dh2
            oh     =  hco3t
            !
            !**** temperature adjustment
            akw = 10.0**(-283.971-
0.05069842*t1k+13323.0/t1k+102.24447*log10(t1k)-1119669.0/(t1k*t1k))/oh
            ak1 = 10.0**(-3404.71/t1k+14.8435-0.032786*t1k)*h2co3t/hco3t
            ak2 = 10.0**(-2902.39/t1k+ 6.4980-0.023790*t1k)*hco3t/co3t
            !
            !**** ph evaluation
            pht = - c(n,k,I_pH) - 2.1
            If (c(n,k,I_pH) <= 0.0) pht = -14.0
            incr = 10.0
            Do ni = 1,3
                  fph    = 1.0
                  incr = incr/10.0
                  iter = 0
                  Do while (fph > 0.0 .and. iter < 12)
                    pht    = pht + incr
                    hion   = 10.0**pht
                    bicart = cart*ak1*hion/(ak1*hion+ak1*ak2+hion*hion)
```

```
                  fph      = bicart*(hion+2.0*ak2)/hion+akw/hion-alkt-
hion/oh
                  iter   = iter+1
                End Do
                pht = pht - incr
            End Do
            !
            !**** ph, carbon dioxide, bicarbonate, and carbonate
concentrations
            hion      =  10.0**pht
            c(n,k,I_pH) = -pht
            CO2         =  c(n,k,I_TIOC)/(1.0 + ak1/hion +
ak1*ak2/(hion*hion))
            HCO3        =  c(n,k,I_TIOC)/(1.0 + hion/ak1+ak2/hion)
            CO3         =  c(n,k,I_TIOC)/((hion*hion)/(ak1*ak2) +
hion/ak2 + 1.0)

      Return


Return
End Subroutine
```