Code Standards and Run-Time configurable WW3

Denise Worthen (Lynker/EMC/NCEP)
Mariana Vertenstein (NCAR)

Currently WW3 uses #ifdefs to control physics choices or other configurable features. In modern F90 code, #ifdefs are largely avoided because of their intrinsic downsides. Not only do they impose compile-time constraints on the run-time execution, they also make code difficult to read, understand and extend. These intrinsic downsides help explain why #ifdefs are avoided in other current earth system components. Removal of #ifdefs from WW3 would provide the development community with a more modern, robust and easily maintained code base and it would allow WW3 to begin the transition to a fully run-time configurable code.

A project to convert #ifdefs to logical flags has been initiated by the authors using a python script, which allows the procedure to be largely automated. The script also provides an opportunity to apply uniform F90 indenting and syntax automatically. The authors would like to engage the WW3 developer community in order to get feedback and gauge support for this project

Proposals for Code Standards (Discussion 551)

Free Format F90

- no reserved columns; code can begin in column 1
- line length 80-100; shorter is better as long as readability is preserved

Standard indenting

- If-endif and do-endo column alignment (end-if and end-do will have comments for longer blocks)
- Comments align with code blocks

Implement Fortran Best practices

- Include *use, only ::* when accessing module variables
- Use *Implicit none* at module level, *Optional* arguments use keywords
- Initialize pointer arrays in derived types to null(); Pointer arrays that are module variables should be initialized to null(); use of *"if associated()"* to determine if pointers have been allocated instead of #ifdefs
- Remove Fortran instrinsics used as variable names (*write*, *status, form* defined as variables)
- Use Fortran character string intrinsics; use assumed length when strings are passed as arguments
- Use utility routines when possible (e.g. memcheck)
- Remove trailing whitespace (both vim and emacs have auto-tools for this)

Example:

Free Format + Indenting

```fortran
module mod1

  use module A, only : var1

  ! module default
  implicit none

  public

contains

  ! this describes the subroutine
  subroutine sr1()

    ! this is a use statement
    use module B, only : somevariable

    ! these are subroutine arguments
    integer, intent(in)  :: invar
    integer, intent(out) :: outvar
    logical, intent(in), optional :: flag

    ! these are local variables
    integer :: jindex
    real :: var1

    ! this is a do loop comment
    do j = 1,jmax
       ! this is an interior comment
       if (somecondition) then
          call a_subroutine_with_large_argument_list(arg1,arg2,arg3,arg4,arg5,arg6,arg7, &
               arg8, arg9)
       end if
    end do

  end subroutine sr1
end module mod1
```

Example: Formatting and Indenting

Existing

```
!
! 1.e Ice concentration interval
!
      IF ( FLICE ) THEN
          IF ( TICE(1) .GE. 0 ) THEN
              DTI0   = DSEC21 ( TICE , TIN )
            ELSE
              DTI0   = 1.
            END IF
#ifdef W3_T
          WRITE (NDST,9014) DTI0
#endif
          IF ( DTI0 .LT. 0. ) THEN
              IF ( IAPROC .EQ. NAPERR ) WRITE (NDSE,1004)
              CALL EXTCDE ( 5 )
            END IF
        ELSE
          DTI0   = 0.
        END IF
#ifdef W3_DEBUGINIT
      WRITE(740+IAPROC,*) 'W3WAVE, step 6'
      FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
#ifdef W3_DEBUGCOH
          CALL ALL_VA_INTEGRAL_PRINT(IMOD, "W3WAVEMD, step 6")
#endif
#endif
```

Aligned

```
          !
          ! 1.e Ice concentration interval
          !
          IF ( FLICE ) THEN
             IF ( TICE(1) .GE. 0 ) THEN
                DTI0   = DSEC21 ( TICE , TIN )
             ELSE
                DTI0   = 1.
             END IF
#ifdef W3_T
             WRITE (NDST,9014) DTI0
#endif
             IF ( DTI0 .LT. 0. ) THEN
                IF ( IAPROC .EQ. NAPERR ) WRITE (NDSE,1004)
                CALL EXTCDE ( 5 )
             END IF
          ELSE
             DTI0   = 0.
          END IF
#ifdef W3_DEBUGINIT
          WRITE(740+IAPROC,*) 'W3WAVE, step 6'
          FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
#ifdef W3_DEBUGCOH
          CALL ALL_VA_INTEGRAL_PRINT(IMOD, "W3WAVEMD, step 6")
#endif
#endif
```

Example: Use, Only

Existing

```
!/ ------------------------
      USE CONSTANTS
!/
      USE W3GDATMD
      USE W3WDATMD
      USE W3ADATMD
      USE W3IDATMD
      USE W3ODATMD
!/
```

>> 100 variables in W3GDATMD !

Specified: User (and compiler) can determine what is actually needed, used and potentially changed in w3gdatmd variables

```
 USE W3GDATMD    , only : IGRID, NSEAL, NSPEC, NX, NY, NK, NSEA
 USE W3GDATMD    , only : GTYPE, UNGTYPE, SMCTYPE, RSTYPE, FILEXT
 USE W3GDATMD    , only : MAPSF, MAPFS, MAPSTA, IOBP, CTHG0S
 USE W3GDATMD    , only : FLCTH, FSREFRACTION, FLCK, FSFREQSHIFT, FLAGLL, FLDRY
 USE W3GDATMD    , only : FSTOTALIMP, FLCX, FLCY, FLSOU, FLAGST
 USE W3GDATMD    , only : SIG, CLATS, TRNX, TRNY
 USE W3GDATMD    , only : DTMAX, DTCFLI, DTH, DMIN
 USE W3GDATMD    , only : XGRD, YGRD, ZB
 USE W3GDATMD    , only : NITERSEC1
```

```
#ifdef W3_REF1
      USE W3GDATMD    , only : RLGTYPE, SX, SY, CLGTYPE, HPFAC, HQFAC, REFLC, REFLD
#endif
```

```
#ifdef W3_IC3
      USE W3GDATMD    , only : IC3PARS
      USE W3SIC3MD    , only : CALLEDIC3TABLE, IC3TABLE_CHENG, W3IC3WNCG_V1, W3IC3WNCG_CHENG
      USE W3IDATMD    , only : ICEP1, ICEP2, ICEP3, ICEP4
#endif
```

Example: Poor Fortran use

Fortran intrinsics used as variables

```
      LOGICAL                  :: WRITE
      CHARACTER(LEN=3)         :: TSFLD
      CHARACTER(LEN=11)        :: FORM = 'UNFORMATTED'
```

```
logical :: write_flag
logical :: form_type
```

Strings not passed with assumed length

```
    CHARACTER, INTENT(IN)              :: INXOUT*(*)
    CHARACTER, INTENT(IN), OPTIONAL    :: FEXT*(*), FPRE*(*)
```

```
character(len=*), intent(in) :: inxout
character(len=*), intent(in) :: fext
character(len=*), intent(in) :: fpre
```

Simple utility routines not implemented

```
#ifdef W3_MEMCHECK
       write(40000+IAPROC,*) 'memcheck_____:', 'WW3_WAVE'
       call getMallocInfo(mallinfos)
       call printMallInfo(IAPROC+40000,mallInfos)
#endif
```

```
call print_memcheck(.....)
```

Disruptive impact

- Majority of changes will be in white space
- Line changes will occur when short lines are extended
- Code changes will occur in declarations (use, only)
- Code changes will occur where intrinsics are re-defined etc

Proposal to remove #ifdefs (763)

- Not all code is compiled. Testing of all possible permutations of #ifdefs is not possible.
  - Many ifdefs only make sense when other #ifdefs are also set and conflict when a third #ifdef is set.
- Compile-time dependence of features or physics options
  - Code must be recompiled to exercise optional features. Makes code rigid.
- Poor Code Readability.
  - Developers cannot easily determine which section of code will execute.
  - Code becomes unnecessarily long when overused or used when not needed.
- Developers are incentivized to "wall off" their changes instead of utilizing more extensible code.
  - Passing an argument under #ifdef control instead of using the optional construct.

➔ #ifdefs should be used only when required
➔ #ifdef removal yields better tested code
➔ #ifdef removal yields run-time configuration

Issues with current #ifdef use

- Used when unnecessary

```
#ifdef W3_MPI
      CALL W3MPIO ( IMOD )
#endif
```

W3MPIO contains 557 *#ifdef W3_MPI* statements!

```
!
! 2.d Save locally stored results
!
#ifdef W3_MPI
      DO JSEA=1, NSEAL
        CALL INIT_GET_ISEA(ISEA, JSEA)
        IXY    = MAPSF(ISEA,3)
        IF (MAPSTA(IXY) .NE. 0) A(ISPEC,JSEA) = SSTOR
        END DO
#endif
!
! 2.e Check if any sends have finished
!
#ifdef W3_MPI
      IB0    = IBFLOC
#endif
!
#ifdef W3_MPI
```

Serial *#ifdef W3_MPI*

Issues with current #ifdef use

- #ifdefs used in argument lists

```
!/ ------------------------------------------------------------------
      SUBROUTINE W3WAVE ( IMOD, ODAT, TEND, STAMP, NO_OUT &
#ifdef W3_OASIS
                  ,ID_LCOMM, TIMEN                 &
#endif
                        )
```

```
                CALL W3SRCE(srce_imp_pre, IT, ISEA, JSEA, IX, IY, IMOD, &
                   VAold(:,JSEA), VA(:,JSEA),                        &
                   VSioDummy, VDioDummy, SHAVETOT(JSEA),  &
                   ALPHA(1:NK,JSEA), WN(1:NK,ISEA),                &
                   CG(1:NK,ISEA), CLATS(ISEA), DW(ISEA), U10(ISEA),            &
                   U10D(ISEA),                                     &
#ifdef W3_FLX5
                   TAUA(ISEA), TAUADIR(ISEA),                  &
#endif
                   AS(ISEA), UST(ISEA),               &
```

Arguments should be passed as keyworded optional argument

Issues with current #ifdef use

- #ifdefs without matching #endif

```
#ifdef W3_SHRD
        IF ( .NOT. FLRBPI(I) .AND. FLBPI ) THEN
#endif
#ifdef W3_MPI
        IF ( .NOT. FLRBPI(I) .AND. FLBPI .AND.                    &
              MPI_COMM_GRD .NE. MPI_COMM_NULL) THEN
#endif
            CALL WMUSET ( MDSE, MDST, NDS(9), .FALSE. )
            IF ( BCDUMP(I) .AND. IAPROC.EQ.NAPBPT ) THEN
                J            = LEN_TRIM(FILEXT)
                TNAME(1:5)   = 'nest.'
                TNAME(6:5+J) = FILEXT(1:J)
                J      = J + 5
                CALL WMUGET ( MDSE, MDST, NDS(9), 'OUT' )
                CALL WMUSET ( MDSE, MDST, NDS(9), .TRUE.,             &
                              NAME=TRIM(FNMPRE)//TNAME(1:J),           &
                              DESC='Output data file (nest dump)' )
                MDS(9,I) = NDSFND
              ELSE
                NDS(9) = -1
              END IF
          END IF
```

Final ENDIF matches either *W3_SHRD* or *W3_MPI*

- emacs can find these

Issues with current #ifdef use

- #ifdefs used in branching if-then blocks

```
!
! 4.d Perform output
!
#ifdef W3_NL5
              IF (J .EQ. 2) TOSNL5  = TONEXT(:, 2)
#endif
                  TOUT(:) = TONEXT(:,J)
                  DTTST   = DSEC21 ( TIME, TOUT )
!
                  IF ( DTTST .EQ. 0. ) THEN
                      IF ( ( J .EQ. 1 )              &
#ifdef W3_SBS
                           .OR. ( J .EQ. 7 )         &
#endif
                                          ) THEN
                         IF ( IAPROC .EQ. NAPFLD ) THEN
```

#ifdefs should only be used with fully-contained if-endif blocks:

```
#ifdef option
   if (do-something) then
   end if
#endif
```

Issues with current #ifdef use

- excessive granularity of "debug" #ifdefs obscures actual code statements and calls

```
!
! 3.1 Interpolate winds, currents, and momentum.
!     (Initialize wave fields with winds)
!
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'FLCUR=', FLCUR
        FLUSH(740+IAPROC)
#endif
#ifdef W3_DEBUGDCXDX
       WRITE(740+IAPROC,*) 'Debug DCXDX FLCUR=', FLCUR
#endif
#ifdef W3_MEMCHECK
       write(40000+IAPROC,*) 'memcheck_____:', 'WW3_WAVE TIME LOOP 3a '
       call getMallocInfo(mallinfos)
       call printMallInfo(IAPROC+40000,mallInfos)
#endif

        IF ( FLCUR  ) THEN
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4'
        FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
#ifdef W3_DEBUGCOH
          CALL ALL_VA_INTEGRAL_PRINT(IMOD, "Before UCUR")
#endif
#endif
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.1'
        FLUSH(740+IAPROC)
#endif
#ifdef W3_TIMINGS
         CALL PRINT_MY_TIME("W3WAVE, step 6.4.1")
#endif

#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.2 before W3UCUR'
        FLUSH(740+IAPROC)
#endif
         CALL W3UCUR ( FLFRST )
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.1 after W3UCUR'
        FLUSH(740+IAPROC)
#endif
```

```
      IF ( FL_ALL ) THEN
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8'
    FLUSH(740+IAPROC)
#endif
          ALLOCATE ( WDATAS(IMOD)%VA(NSPEC,0:NSEALM), STAT=ISTAT ); WDATA
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.1'
    FLUSH(740+IAPROC)
#endif
          CHECK_ALLOC_STATUS ( ISTAT )
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.2'
    FLUSH(740+IAPROC)
#endif
!!/PDLIB          ALLOCATE ( WDATAS(IMOD)%VAOLD(NSPEC,0:NSEALM) )
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.3'
    FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
          ALLOCATE ( WDATAS(IMOD)%SHAVETOT(NSEAL), stat=istat )
#endif
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.4, stat=', istat
    FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
        IF (.not. LSLOC) THEN
          ALLOCATE ( WDATAS(IMOD)%VSTOT(NSPEC,NSEAL), stat=istat )
#endif
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.5, stat=', istat
    FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
          ALLOCATE ( WDATAS(IMOD)%VDTOT(NSPEC,NSEAL), stat=istat )
#endif
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.6, stat=', istat
    FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
        ENDIF ! LSLOC
          ALLOCATE ( WDATAS(IMOD)%VAOLD(NSPEC,NSEAL), stat=istat )
#endif
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.7, stat=', istat
    FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
        IF (.not. LSLOC) THEN
         WDATAS(IMOD)%VSTOT=0
#endif
#ifdef W3_DEBUGINIT
    WRITE(740+IAPROC,*) 'W3DIMW, step 8.8'
    FLUSH(740+IAPROC)
#endif
```

Proposed solutions to #ifdef issues

Substitute logical flags and utility routines

Initialize flags at compile time

```
#ifdef W3_DEBUGINIT
   logical, parameter :: w3_debuginit_flag = .true.
#else
   logical, parameter :: w3_debuginit_flag = .false
#endif
```

```
#ifdef W3_DEBUGINIT
   do stuff
#endif
```

```
if (w3_debuginit_flag) then
   do stuff
end if
```

Add new utility routines

```
  !=====================================================================
  !> Write memory statistics if requested
  !!
  !> @details Writes a single line of memory statistics
  !!
  !! @param[in]   iun               unit number
  !! @param[in]   msg               message
  !!
  !> @author mvertens@ucar.edu, Denise.Worthen@noaa.gov
  !> @date 06-01-2022

  subroutine print_memcheck(iun, msg)
#if W3_MEMCHECK
    USE MallocInfo_m
#endif
    integer          , intent(in) :: iun
    character(len=*) , intent(in) :: msg

#if W3_MEMCHECK
    write(iun,*) trim(msg)
    call getMallocInfo(mallinfos)
    call printMallInfo(iun, mallInfos)
#endif
  end subroutine print_memcheck
```

```
call print_memcheck(IAPROC+40000, 'memcheck_____:'//' WW3_WAVE TIME LOOP 4')
```

➔ Flags eventually set via namelist, allowing run-time configuration for most configurations and options

Use extensible constructs instead of branching if-endifs

```
!
! 4.d Perform output
!
#ifdef W3_NL5
              IF (J .EQ. 2) TOSNL5  = TONEXT(:, 2)
#endif
                  TOUT(:) = TONEXT(:,J)
                  DTTST   = DSEC21 ( TIME, TOUT )
!
                  IF ( DTTST .EQ. 0. ) THEN
                      IF ( ( J .EQ. 1 )              &
#ifdef W3_SBS
                          .OR. ( J .EQ. 7 )          &
#endif
                                          ) THEN
                        IF ( IAPROC .EQ. NAPFLD ) THEN
```

```
                    !
                    ! Determine output flags
                    !
                    if (w3_sbs_flag) then
                       do_gridded_output = ( j .eq. 1 )  .or. ( j .eq. 7 )
                    else
                       do_gridded_output = ( j .eq. 1 )
                    end if

                    !
                    ! 4.d Perform output
                    !
#ifdef W3_NL5
              IF (J .EQ. 2) TOSNL5  = TONEXT(:, 2)
#endif
                    TOUT(:) = TONEXT(:,J)
                    DTTST   = DSEC21 ( TIME, TOUT )
!
                    IF ( DTTST .EQ. 0. ) THEN
                       if (do_gridded_output) then
```

Reduce Granularity of "debug" ifdefs and serial #ifdefs

- retain only at the enter/exit stage of a subroutine and at the enter/exit stage of major code blocks

```
!
! 3.1 Interpolate winds, currents, and momentum.
!     (Initialize wave fields with winds)
!
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'FLCUR=', FLCUR
        FLUSH(740+IAPROC)
#endif
#ifdef W3_DEBUGDCXDX
       WRITE(740+IAPROC,*) 'Debug DCXDX FLCUR=', FLCUR
#endif
#ifdef W3_MEMCHECK
       write(40000+IAPROC,*) 'memcheck_____:', 'WW3_WAVE TIME LOOP 3a '
       call getMallocInfo(mallinfos)
       call printMallInfo(IAPROC+40000,mallInfos)
#endif

          IF ( FLCUR  ) THEN
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4'
        FLUSH(740+IAPROC)
#endif
#ifdef W3_PDLIB
#ifdef W3_DEBUGCOH
          CALL ALL_VA_INTEGRAL_PRINT(IMOD, "Before UCUR")
#endif
#endif
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.1'
        FLUSH(740+IAPROC)
#endif
#ifdef W3_TIMINGS
         CALL PRINT_MY_TIME("W3WAVE, step 6.4.1")
#endif

#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.2 before W3UCUR'
        FLUSH(740+IAPROC)
#endif
          CALL W3UCUR ( FLFRST )
#ifdef W3_DEBUGRUN
        WRITE(740+IAPROC,*) 'W3WAVE, step 6.4.1 after W3UCUR'
        FLUSH(740+IAPROC)
```

```
      !
      ! 3.1 Interpolate winds, currents, and momentum.
      !     (Initialize wave fields with winds)
      !
      if (w3_debugrun_flag) then
         WRITE(740+IAPROC,*) 'Starting 3.1, Interpolate winds, currents, and momentum'
      end if
      call print_memcheck(memunit, 'memcheck_____:'// ' WW3_WAVE TIME LOOP 3a ')

      IF ( FLCUR  ) THEN
#ifdef W3_PDLIB
               if (w3_debugcoh_flag) then
                  CALL ALL_VA_INTEGRAL_PRINT(IMOD, "Before UCUR")
               end if
#endif
               if (w3_timings_flag) then
                  CALL PRINT_MY_TIME("W3WAVE, step 6.4.1")
               end if

               CALL W3UCUR ( FLFRST )
               call print_memcheck(memunit, 'memcheck_____:'// ' WW3_WAVE TIME LOOP 3b ')


               IF (GTYPE .EQ. SMCTYPE) THEN
                  IX = 1
#ifdef W3_SMC
 !!Li  Use new sub for DCXDX/Y and DCYDX/Y assignment.
                  CALL SMCDCXY
#endif
               ELSE IF (GTYPE .EQ. UNGTYPE) THEN
```

Remove other unnecessary #ifdefs

- most module variables
- definitions of derived types
- local variables
- format statements
- around OpenMP blocks

Disruptive impact

- Large, but changes can be staged
- Argument for "status quo" --- too many other issues/problems/development on the horizon
- Counter-argument is that dealing with issues/problems/development is **easier, faster** and **more robust** with clean, well-tested code

Final thoughts:

➔ Deferred action on addressing these issues has only made the problem grow over time.

➔ Pain will be necessary at some point, further delaying the inevitable is not wise.