GEOS-Chem Reference, Vol. 2: Utility Modules

Bob Yantosca, Claire Carouge, and Philippe Le Sager
School of Engineering and Applied Sciences, Harvard University

December 16, 2010

Contents

1 Routine/Function Prologues

1.1 Fortran: Module Interface bpch2_mod

Module BPCH2_MOD contains the routines used to read data from and write data to binary punch (BPCH) file format (v. 2.0).

INTERFACE:

```
MODULE BPCH2_MOD
```

USES:

```
IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  :: OPEN_BPCH2_FOR_READ
PUBLIC  :: OPEN_BPCH2_FOR_WRITE
PUBLIC  :: BPCH2_HDR
PUBLIC  :: BPCH2
PUBLIC  :: READ_BPCH2
PUBLIC  :: GET_MODELNAME
PUBLIC  :: GET_NAME_EXT
PUBLIC  :: GET_NAME_EXT_2D
PUBLIC  :: GET_RES_EXT
PUBLIC  :: GET_HALFPOLAR
PUBLIC  :: GET_TAU0

INTERFACE GET_TAU0
   MODULE PROCEDURE GET_TAU0_6A
END INTERFACE
```

PRIVATE MEMBER FUNCTIONS:

```
PRIVATE :: GET_TAU0_6A
```

REVISION HISTORY:

```
(1 ) Added routine GET_TAU0 (bmy, 7/20/00)
(2 ) Added years 1985-2001 for routine GET_TAU0 (bmy, 8/1/00)
(3 ) Use IOS /= 0 criterion to also check for EOF (bmy, 9/12/00)
(4 ) Removed obsolete code in "read_bpch2.f" (bmy, 12/18/00)
(5 ) Correct error for 1991 TAU values in GET_TAU0 (bnd, bmy, 1/4/01)
(6 ) BPCH2_MOD is now independent of any GEOS-CHEM size parameters.
      (bmy, 4/18/01)
(7 ) Now have 2 versions of "GET_TAU0" overloaded by an interface.  The
      original version takes 2 arguments (MONTH, YEAR).  The new version
      takes 3 arguments (MONTH, DAY, YEAR). (bmy, 8/22/01)
```

```
    (8 ) Updated comments (bmy, 9/4/01)
    (9 ) Renamed GET_TAU0_3A to GET_TAU0_6A, and updated the GET_TAU0
          interface.  Also updated comments (bmy, 9/26/01)
    (10) Now use special model name for GEOS-3 w/ 30 layers (bmy, 10/9/01)
    (11) Minor bug fix in GET_TAU0_2A.  Also deleted obsolete code from 9/01.
          (bmy, 11/15/01)
    (12) Moved routines JULDAY, MINT, CALDATE to "julian_mod.f".  Now
          references routine JULDAY from "julday_mod.f".  Also added code
          for GEOS-4/fvDAS model type. (bmy, 11/20/01)
    (23) Now divide module header into MODULE PRIVATE, MODULE VARIABLES, and
          MODULE ROUTINES sections.  Also add MODULE INTERFACES section,
          since we have an interface here. (bmy, 5/28/02)
    (24) Added OPEN_BPCH2_FOR_READ and OPEN_BPCH2_FOR_WRITE.  Also now
          reference IU_FILE and IOERROR from "file_mod.f". (bmy, 7/30/02)
    (25) Now references "error_mod.f".  Also obsoleted routine GET_TAU0_2A.
          (bmy, 10/15/02)
    (26) Made modification in READ_BPCH2 for 1x1 nested grids (bmy, 3/11/03)
    (27) Modifications for GEOS-4, 30-layer grid (bmy, 11/3/03)
    (28) Added cpp switches for GEOS-4 1x125 grid (bmy, 12/1/04)
    (29) Modified for GCAP and GEOS-5 met fields.  Added function
          GET_HALFPOLAR. (bmy, 6/28/05)
    (30) Added GET_NAME_EXT_2D to get filename extension for files which do
          not contain any vertical information (bmy, 8/16/05)
    (31) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
    (32) Renamed GRID30LEV to GRIDREDUCED.  Also increase TEMPARRAY in
          READ_BPCH2 for GEOS-5 vertical levels. (bmy, 2/16/07)
    (33) Modifications for GEOS-5 nested grids (bmy, 11/6/08)
    20 Nov 2009 - R. Yantosca - Added ProTeX headers
    13 Aug 2010 - R. Yantosca - Added modifications for MERRA
```

1.1.1 open_bpch2_for_read

Subroutine OPEN_BPCH2_FOR_READ opens a binary punch file (version 2.0 format) for reading only. Also reads FTI and TITLE strings.

INTERFACE:

```
      SUBROUTINE OPEN_BPCH2_FOR_READ( IUNIT, FILENAME, TITLE )
```

USES:

```
      USE ERROR_MOD, ONLY : ERROR_STOP
      USE FILE_MOD,  ONLY : IOERROR
```

INPUT PARAMETERS:

```
      INTEGER,           INTENT(IN)            :: IUNIT     ! LUN for file I/O
      CHARACTER(LEN=*),  INTENT(IN)            :: FILENAME  ! Name of file
```

OUTPUT PARAMETERS:

```
CHARACTER(LEN=80), INTENT(OUT), OPTIONAL :: TITLE     ! File title string
```

REVISION HISTORY:

```
(1 ) Now references ERROR_STOP from "error_mod.f" (bmy, 10/15/02)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.1.2 open_bpch2_for_write

Subroutine OPEN_BPCH2_FOR_WRITE opens a binary punch file (version 2.0) for writing.

INTERFACE:

```
SUBROUTINE OPEN_BPCH2_FOR_WRITE( IUNIT, FILENAME, TITLE )
```

USES:

```
USE FILE_MOD, ONLY : IOERROR
```

INPUT PARAMETERS:

```
INTEGER,           INTENT(IN)           :: IUNIT     ! LUN for file I/O
CHARACTER(LEN=*),  INTENT(IN)           :: FILENAME  ! Name of file
```

OUTPUT PARAMETERS:

```
CHARACTER(LEN=80), INTENT(OUT), OPTIONAL :: TITLE     ! File title string
```

REVISION HISTORY:

```
30 Jul 2002 - R. Yantosca - Initial version
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.1.3 bpch2_hdr

Subroutine BPCH2_HDR writes a header at the top of the binary punch file, version 2.0.

INTERFACE:

```
SUBROUTINE BPCH2_HDR ( IUNIT, TITLE )
```

USES:

```
USE FILE_MOD, ONLY : IOERROR
```

INPUT PARAMETERS:

```
INTEGER,           INTENT(IN) :: IUNIT   ! LUN for file I/O
CHARACTER(LEN=80), INTENT(IN) :: TITLE   ! Top-of-file title string
```

REVISION HISTORY:

```
(1 ) Added this routine to "bpch_mod.f" (bmy, 6/28/00)
(2 ) Use IOS /= 0 criterion to also check for EOF condition (bmy, 9/12/00)
(3 ) Now reference IOERROR from "file_mod.f". (bmy, 6/26/02)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.1.4 bpch2

Subroutine BPCH2 writes binary punch file (version 2.0) to disk. Information about the model grid is also stored with each data block.

INTERFACE:

```
       SUBROUTINE BPCH2( IUNIT,     MODELNAME, LONRES,   LATRES,
     &                  HALFPOLAR, CENTER180, CATEGORY, NTRACER,
     &                  UNIT,      TAU0,      TAU1,     RESERVED,
     &                  NI,        NJ,        NL,       IFIRST,
     &                  JFIRST,    LFIRST,    ARRAY )
```

USES:

```
       USE FILE_MOD, ONLY : IOERROR
```

INPUT PARAMETERS:

```
       ! Arguments
       INTEGER,           INTENT(IN) :: IUNIT            ! LUN for file I/O
       CHARACTER(LEN=20), INTENT(IN) :: MODELNAME        ! Met field type
       REAL*4,            INTENT(IN) :: LONRES           ! Lon resolution [deg]
       REAL*4,            INTENT(IN) :: LATRES           ! Lat resolution [deg]
       INTEGER,           INTENT(IN) :: HALFPOLAR        ! 1/2-size polar boxes?
       INTEGER,           INTENT(IN) :: CENTER180        ! 1st box center -180?
       CHARACTER(LEN=40), INTENT(IN) :: CATEGORY         ! Diag. category name
       INTEGER,           INTENT(IN) :: NTRACER          ! Tracer index #
       CHARACTER(LEN=40), INTENT(IN) :: UNIT             ! Unit string
       REAL*8,            INTENT(IN) :: TAU0             ! TAU values @ start &
       REAL*8,            INTENT(IN) :: TAU1             !  end of diag interval
       CHARACTER(LEN=40), INTENT(IN) :: RESERVED         ! Extra string
       INTEGER,           INTENT(IN) :: NI, NJ, NL       ! Dimensions of ARRAY
       INTEGER,           INTENT(IN) :: IFIRST           ! (I,J,L) indices of
       INTEGER,           INTENT(IN) :: JFIRST           !  the first grid box
       INTEGER,           INTENT(IN) :: LFIRST           !  in Fortran notation
       REAL*4,            INTENT(IN) :: ARRAY(NI,NJ,NL)  ! Data array
```

REVISION HISTORY:

```
    (1 ) Added indices to IOERROR calls (e.g. "bpch2:1", "bpch2:2", etc.)
          (bmy, 10/4/99)
    (2 ) Added this routine to "bpch_mod.f" (bmy, 6/28/00)
    (3 ) Use IOS /= 0 criterion to also check for EOF condition (bmy, 9/12/00)
    (4 ) Now reference IOERROR from "file_mod.f". (bmy, 6/26/02)
```

1.1.5 read_bpch2

Subroutine READ_BPCH2 reads a binary punch file (v. 2.0) and extracts a data block that matches the given category, tracer, and tau value.

INTERFACE:

```
      SUBROUTINE READ_BPCH2( FILENAME, CATEGORY_IN, TRACER_IN,
     &                       TAU0_IN,  IX,          JX,
     &                       LX,       ARRAY,       QUIET )
```

USES:

```
      USE ERROR_MOD, ONLY : ERROR_STOP
      USE FILE_MOD,  ONLY : IU_FILE, IOERROR

#     include "define.h"
```

INPUT PARAMETERS:

```
      CHARACTER(LEN=*),  INTENT(IN)  :: FILENAME        ! Bpch file to read
      CHARACTER(LEN=*),  INTENT(IN)  :: CATEGORY_IN     ! Diag. category name
      INTEGER,           INTENT(IN)  :: TRACER_IN       ! Tracer index #
      REAL*8,            INTENT(IN)  :: TAU0_IN         ! TAU timestamp
      INTEGER,           INTENT(IN)  :: IX, JX, LX      ! Dimensions of ARRAY
      LOGICAL, OPTIONAL, INTENT(IN)  :: QUIET           ! Don't print output
```

OUTPUT PARAMETERS:

```
      REAL*4,            INTENT(OUT) :: ARRAY(IX,JX,LX)  ! Data array from file
```

REVISION HISTORY:

```
  (1 ) Assumes that we are reading in a global-size data block.
  (2 ) Trap all I/O errors with subroutine IOERROR.F.
  (3 ) Now stop with an error message if no matches are found. (bmy, 3/9/00)
  (4 ) Added this routine to "bpch_mod.f" (bmy, 6/28/00)
  (5 ) Use IOS /= 0 criterion to also check for EOF condition (bmy, 9/12/00)
  (6 ) TEMPARRAY now dimensioned to be of global size (bmy, 10/12/00)
  (7 ) Removed obsolete code from 10/12/00 (bmy, 12/18/00)
  (8 ) Now make TEMPARRAY independent of CMN_SIZE parameters (bmy, 4/17/01)
  (9 ) Removed old commented-out code (bmy, 4/20/01)
  (10) Now reference IU_FILE and IOERROR from "file_mod.f".  Now call
        OPEN_BPCH2_FOR_READ to open the binary punch file.  Now use IU_FILE
        as the unit number instead of a locally-defined IUNIT. (bmy, 7/30/02)
  (11) Now references ERROR_STOP from "error_mod.f" (bmy, 10/15/02)
  (12) Now set IFIRST=1, JFIRST=1 for 1x1 nested grids.  Now needs to
        reference "define.h".  Added OPTIONAL QUIET flag. (bmy, 3/14/03)
  (13) Now separate off nested grid code in an #ifdef block using
        NESTED_CH or NESTED_NA cpp switches (bmy, 12/1/04)
  (14) Make TEMPARRAY big enough for GEOS-5 72 levels (and 73 edges)
        (bmy, 2/15/07)
  (15) Make TEMPARRAY large enough for 0.5 x 0.666 arrays -- but only if we
        are doing a 0.5 x 0.666 nested simulation. (yxw, dan, bmy, 11/6/08)
  20 Nov 2009 - R. Yantosca - Added ProTeX header
  18 Dec 2009 - Aaron van D - Add NESTED_EU flag
```

1.1.6 get_modelname

Function GET_MODELNAME returns the proper value of MODELNAME for current GEOS or GCAP met field type. MODELNAME is written to the binary punch file and is also used by the GAMAP package.

INTERFACE:

```
      FUNCTION GET_MODELNAME() RESULT( MODELNAME )
```

USES:

```
#     include "CMN_SIZE"
```

RETURN VALUE:

```
      CHARACTER(LEN=20) :: MODELNAME   ! Model name for the current met field
```

REVISION HISTORY:

```
    (1 ) Now use special model name for GEOS-3 w/ 30 layers (bmy, 10/9/01)
    (2 ) Added modelname for GEOS-4/fvDAS model type (bmy, 11/20/01)
    (3 ) Added "GEOS4_30L" for reduced GEOS-4 grid.  Also now use C-preprocessor
          switch "GRID30LEV" instead of IF statements. (bmy, 11/3/03)
    (4 ) Updated for GCAP and GEOS-5 met fields.  Rearranged coding for
          simplicity. (swu, bmy, 5/24/05)
    (5 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
    (6 ) Rename GRID30LEV to GRIDREDUCED (bmy, 2/7/07)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
    13 Aug 2010 - R. Yantosca - Added MERRA model names
```

1.1.7 get_name_ext

Function GET_NAME_EXT returns the proper filename extension the current GEOS-Chem met field type (e.g. "geos3", "geos4", "geos5", or "gcap").

INTERFACE:

```
      FUNCTION GET_NAME_EXT() RESULT( NAME_EXT )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
#if   defined( GEOS_3 )
      CHARACTER(LEN=5) :: NAME_EXT
      NAME_EXT = 'geos3'

#elif defined( GEOS_4 )
      CHARACTER(LEN=5) :: NAME_EXT
      NAME_EXT = 'geos4'

#elif defined( GEOS_5 )
      CHARACTER(LEN=5) :: NAME_EXT
      NAME_EXT = 'geos5'

#elif defined( GCAP )
      CHARACTER(LEN=4) :: NAME_EXT
      NAME_EXT = 'gcap'

#elif defined( MERRA )
      CHARACTER(LEN=5) :: NAME_EXT
      NAME_EXT = 'geos5'

#endif
```

REVISION HISTORY:

```
(1 ) Added name string for GEOS-4/fvDAS model type (bmy, 11/20/01)
(2 ) Remove obsolete "geos2" model name strning (bmy, 11/3/03)
(3 ) Modified for GCAP and GEOS-5 met fields (bmy, 5/24/05)
(4 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
20 Nov 2009 - R. Yantosca - Added ProTeX header
13 Aug 2010 - R. Yantosca - MERRA uses "geos5" name extension since its
                            grid is identical to GEOS-5.
```

1.1.8 get_name_ext_2d

Function GET_NAME_EXT_2D returns the proper filename extension for CTM model name for files which do not contain any vertical information (i.e. "geos" or "gcap").

INTERFACE:

```
      FUNCTION GET_NAME_EXT_2D() RESULT( NAME_EXT_2D )
```

RETURN VALUE:

```
      CHARACTER(LEN=4) :: NAME_EXT_2D
```

REVISION HISTORY:

```
(1 ) Added name string for GEOS-4/fvDAS model type (bmy, 11/20/01)
(2 ) Remove obsolete "geos2" model name strning (bmy, 11/3/03)
(3 ) Modified for GCAP and GEOS-5 met fields (bmy, 5/24/05)
(4 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.1.9 get_res_ext

Function GET_RES_EXT returns the proper filename extension for the GEOS-Chem horizontal grid resolution (e.g. "1x1", "2x25", "4x5").

INTERFACE:

```
      FUNCTION GET_RES_EXT() RESULT( RES_EXT )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
#if   defined( GRID4x5 )
      CHARACTER(LEN=3) :: RES_EXT
      RES_EXT = '4x5'

#elif defined( GRID2x25 )
      CHARACTER(LEN=4) :: RES_EXT
      RES_EXT = '2x25'

#elif defined( GRID1x125 )
      CHARACTER(LEN=5) :: RES_EXT
      RES_EXT = '1x125'

#elif defined( GRID1x1 )
      CHARACTER(LEN=3) :: RES_EXT
      RES_EXT = '1x1'

#elif defined( GRID05x0666 )
      CHARACTER(LEN=7) :: RES_EXT
      RES_EXT = '05x0666'

#endif
```

REVISION HISTORY:

```
   (1 ) Added extension for 1 x 1.25 grid (bmy, 12/1/04)
   (2 ) Added extension for 0.5 x 0.666 grid (yxw, dan, bmy, 11/6/08)
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.1.10 get_halfpolar

Function GET_HALFPOLAR returns 1 if the current grid has half-sized polar boxes (e.g. GEOS), or zero otherwise (e.g. GCAP).

INTERFACE:

```
      FUNCTION GET_HALFPOLAR() RESULT( HALFPOLAR )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
      INTEGER :: HALFPOLAR  ! =1 if we have half-sized polar boxes, =0 if not
```

REVISION HISTORY:

```
   28 Jun 2005 - S. Wu & R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca         - Added ProTeX header
```

1.1.11 get_tau0_6a

Function GET_TAU0_6A returns the corresponding TAU0 value for the first day of a given MONTH of a given YEAR. This is necessary to index monthly mean binary punch files, which are used as input to GEOS-Chem.

This function takes 3 mandatory arguments (MONTH, DAY, YEAR) and 3 optional arguments (HOUR, MIN, SEC). It is intended to replace the current 2-argument version of GET_TAU0. The advantage being that GET_TAU0_6A can compute a TAU0 for any date and time in the GEOS-Chem epoch, rather than just the first day of each month. Overload this w/ an interface so that the user can also choose the version of GET_TAU0 w/ 2 arguments (MONTH, YEAR), which is the prior version.

INTERFACE:

```
      FUNCTION GET_TAU0_6A( MONTH, DAY, YEAR,
     &                      HOUR,  MIN, SEC  ) RESULT( THIS_TAU0 )
```

USES:

```
      USE ERROR_MOD,  ONLY : ERROR_STOP
      USE JULDAY_MOD, ONLY : JULDAY
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN)           :: MONTH
      INTEGER, INTENT(IN)           :: DAY
      INTEGER, INTENT(IN)           :: YEAR
      INTEGER, INTENT(IN), OPTIONAL :: HOUR
      INTEGER, INTENT(IN), OPTIONAL :: MIN
      INTEGER, INTENT(IN), OPTIONAL :: SEC
```

RETURN VALUE:

```
      REAL*8                        :: THIS_TAU0   ! TAU0 timestamp
```

REMARKS:

```
TAU0 is hours elapsed since 00:00 GMT on 01 Jan 1985.
```

REVISION HISTORY:

```
(1 ) 1985 is the first year of the GEOS epoch.
(2 ) Add TAU0 values for years 1985-2001 (bmy, 8/1/00)
(3 ) Correct error for 1991 TAU values.  Also added 2002 and 2003.
      (bnd, bmy, 1/4/01)
(4 ) Updated comments  (bmy, 9/26/01)
(5 ) Now references JULDAY from "julday_mod.f" (bmy, 11/20/01)
(6 ) Now references ERROR_STOP from "error_mod.f"  (bmy, 10/15/02)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.2 Fortran: Module Interface charpak_mod.f

Module CHARPAK_MOD contains routines from the CHARPAK string and character manipulation package used by GEOS-Chem.

INTERFACE:

```
MODULE CHARPAK_MOD
```

USES:

```
IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  :: CNTMAT
PUBLIC  :: COPYTXT
PUBLIC  :: CSTRIP
PUBLIC  :: ISDIGIT
PUBLIC  :: STRREPL
PUBLIC  :: STRSPLIT
PUBLIC  :: STRSQUEEZE
PUBLIC  :: TRANLC
PUBLIC  :: TRANUC
PUBLIC  :: TXT2INUM
PUBLIC  :: TXTEXT
```

REMARKS:

```
CHARPAK routines by Robert D. Stewart, 1992.  Subsequent modifications
made for GEOS-CHEM by Bob Yantosca (1998, 2002, 2004).
```

REVISION HISTORY:

```
(1 ) Moved "cntmat.f", "copytxt.f", "cstrip.f", "fillstr.f", "txt2inum.f",
      "txtext.f", into this F90 module for easier bookkeeping
      (bmy, 10/15/01)
(2 ) Moved "tranuc.f" into this F90 module (bmy, 11/15/01)
(3 ) Now divide module header into MODULE PRIVATE, MODULE VARIABLES, and
      MODULE ROUTINES sections.  Updated comments (bmy, 5/28/02)
(4 ) Wrote a new file "strrepl.f", which replaces a character pattern
      within a string with replacement text.  Moved "tranlc.f" into
      this module.  Replaced calls to function LENTRIM with F90
      intrinsic function LEN_TRIM.  Removed function FILLSTR and
      replaced it w/ F90 intrinsic REPEAT. (bmy, 6/25/02)
(5 ) Added routine STRSPLIT as a wrapper for TXTEXT.  Also added
      routines STRREPL and STRSQUEEZE. (bmy, 7/30/02)
(6 ) Added function ISDIGIT.  Also replace LEN_TRIM with LEN in routine
      STRREPL, to allow us to replace tabs w/ spaces. (bmy, 7/20/04)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.3 Fortran: Module Interface directory_mod.f

Module DIRECTORY_MOD contains the directory path variables used by GEOS-Chem.

INTERFACE:

```
MODULE DIRECTORY_MOD
```

USES:

```
IMPLICIT NONE
PUBLIC
```

PUBLIC DATA MEMBERS:

```
CHARACTER(LEN=255) :: DATA_DIR      ! Main DAO met field directory
CHARACTER(LEN=255) :: DATA_DIR_1x1  ! Root data dir for 1x1 emissions
CHARACTER(LEN=255) :: GCAP_DIR      ! Subdir for GCAP met data
CHARACTER(LEN=255) :: GEOS_1_DIR    ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: GEOS_S_DIR    ! !%%% OBSOLETE
CHARACTER(LEN=255) :: GEOS_3_DIR    ! Subdir for GEOS-3 met data
CHARACTER(LEN=255) :: GEOS_4_DIR    ! Subdir for GEOS-4 met data
CHARACTER(LEN=255) :: GEOS_5_DIR    ! Subdir for GEOS-5 met data
CHARACTER(LEN=255) :: MERRA_DIR     ! Subdir for MERRA  met data
CHARACTER(LEN=255) :: TEMP_DIR      ! Temp dir for unzipping met data
CHARACTER(LEN=255) :: RUN_DIR       ! Run directory for GEOS-Chem
CHARACTER(LEN=255) :: OH_DIR        ! Dir w/ mean OH files
CHARACTER(LEN=255) :: O3PL_DIR      ! Dir w/ archived O3 P/L rate files
CHARACTER(LEN=255) :: TPBC_DIR      ! Dir w/ TPCORE boundary conditions
CHARACTER(LEN=255) :: TPBC_DIR_NA   ! Dir w/ TPCORE BC's for NA nest grid
CHARACTER(LEN=255) :: TPBC_DIR_EU   ! Dir w/ TPCORE BC's for EU nest grid
CHARACTER(LEN=255) :: TPBC_DIR_CH   ! Dir w/ TPCORE BC's for CH nest grid
```

REVISION HISTORY:

```
    20 Jul 2004 - R. Yantosca - Initial version
    25 May 2005 - R. Yantosca - Added variables GCAP_DIR and GEOS_5_DIR
    24 Oct 2005 - R. Yantosca - Added DATA_DIR_1x1
    20 Nov 2009 - R. Yantosca - Added ProTeX header
    18 Dec 2009 - Aaron van D - Added TPBC_DIR_NA, TPBC_DIR_EU, TPBC_DIR_CH
    13 Aug 2010 - R. Yantosca - Added MERRA_DIR for MERRA met fields
```

1.4 Fortran: Module Interface error_mod.f

Module ERROR_MOD contains error checking routines.

INTERFACE:

```
      MODULE ERROR_MOD
```

USES:

```
      IMPLICIT NONE
      PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
      PUBLIC  :: ALLOC_ERR
      PUBLIC  :: CHECK_VALUE
      PUBLIC  :: DEBUG_MSG
      PUBLIC  :: ERROR_STOP
      PUBLIC  :: GEOS_CHEM_STOP
      PUBLIC  :: IS_SAFE_DIV
      PUBLIC  :: IS_SAFE_EXP
      PUBLIC  :: IT_IS_NAN
      PUBLIC  :: IT_IS_FINITE
      PUBLIC  :: SAFE_DIV
      PUBLIC  :: SAFE_EXP
      PUBLIC  :: SAFE_LOG
      PUBLIC  :: SAFE_LOG10

      ! Interface for NaN-check routines
      INTERFACE IT_IS_NAN
         MODULE PROCEDURE NAN_FLOAT
         MODULE PROCEDURE NAN_DBLE
      END INTERFACE

      ! Interface for finite-check routines
      INTERFACE IT_IS_FINITE
         MODULE PROCEDURE FINITE_FLOAT
         MODULE PROCEDURE FINITE_DBLE
      END INTERFACE

      ! Interface for check-value routines
      INTERFACE CHECK_VALUE
         MODULE PROCEDURE CHECK_REAL_VALUE
         MODULE PROCEDURE CHECK_DBLE_VALUE
      END INTERFACE
```

PRIVATE MEMBER FUNCTIONS:

```
      PRIVATE :: CHECK_DBLE_VALUE
      PRIVATE :: CHECK_REAL_VALUE
      PRIVATE :: FINITE_DBLE
      PRIVATE :: FINITE_FLOAT
      PRIVATE :: NAN_DBLE
      PRIVATE :: NAN_FLOAT
```

REVISION HISTORY:

```
    08 Mar 2001 - R. Yantosca - Initial version
    (1 ) Added subroutines CHECK_REAL_VALUE and CHECK_DBLE_VALUE, which are
          overloaded by interface CHECK_VALUE.  This is a convenience
          so that you don't have to always call IT_IS_NAN directly.
          (bmy, 6/13/01)
    (2 ) Updated comments (bmy, 9/4/01)
    (3 ) Now use correct values for bit masking in FINITE_FLOAT for the
          ALPHA platform (bmy, 11/15/01)
    (4 ) Now divide module header into MODULE PRIVATE, MODULE VARIABLES, and
          MODULE ROUTINES sections.  Also add MODULE INTERFACES section,
          since we have an interface here. (bmy, 5/28/02)
    (5 ) Add NaN and infinity error checking for Linux platform (bmy, 3/22/02)
    (6 ) Added routines ERROR_STOP, GEOS_CHEM_STOP, and ALLOC_ERR to this
          module.  Also improved CHECK_STT. (bmy, 11/27/02)
    (7 ) Minor bug fixes in FORMAT statements.   Renamed cpp switch from
          DEC_COMPAQ to COMPAQ.  Also added code to trap errors on SUN
          platform. (bmy, 3/21/03)
    (8 ) Added patches for IBM/AIX platform (gcc, bmy, 6/27/03)
    (9 ) Bug fixes for LINUX platform (bmy, 9/29/03)
    (10) Now supports INTEL_FC compiler (bmy, 10/24/03)
    (11) Changed the name of some cpp switches in "define.h" (bmy, 12/2/03)
    (12) Minor fix for LINUX_IFC and LINUX_EFC (bmy, 1/24/04)
    (13) Do not flush buffer for LINUX_EFC in ERROR_STOP (bmy, 4/6/04)
    (14) Move CHECK_STT routine to "tracer_mod.f" (bmy, 7/20/04)
    (15) Added LINUX_IFORT switch for Intel v8 and v9 compilers (bmy, 10/18/05)
    (16) Now print IFORT error messages for Intel v8/v9 compiler (bmy, 11/30/05)
    (17) Cosmetic change in DEBUG_MSG (bmy, 4/10/06)
    (18) Remove support for LINUX_IFC and LINUX_EFC compilers (bmy, 8/4/06)
    (19) Now use intrinsic functions for IFORT, remove C routines (bmy, 8/14/07)
    (20) Added routine SAFE_DIV (phs, bmy, 2/26/08)
    (21) Added routine IS_SAFE_DIV (phs, bmy, 6/11/08)
    (22) Updated routine SAFE_DIV (phs, 4/14/09)
    (23) Remove support for SGI, COMPAQ compilers (bmy, 7/8/09)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
    04 Jan 2010 - R. Yantosca - Added SAFE_EXP and IS_SAFE_EXP functions
    04 Jan 2010 - R. Yantosca - Added SAVE_LOG and SAFE_LOG10 functions
```

1.4.1 nan_float

Function NAN_FLOAT returns TRUE if a REAL*4 number is equal to the IEEE NaN (Not-a-Number) flag. Returns FALSE otherwise.

INTERFACE:

```
      FUNCTION NAN_FLOAT( VALUE ) RESULT( IT_IS_A_NAN )
```

USES:

```
 #     include "define.h"

 #if   defined( IBM_AIX ) || defined( IBM_XLF )
       USE IEEE_ARITHMETIC
 #endif
```

INPUT PARAMETERS:

```
      REAL*4, INTENT(IN) :: VALUE        ! Value to be tested for NaN
```

RETURN VALUE:

```
      LOGICAL            :: IT_IS_A_NAN  ! =T if VALUE is NaN; =F otherwise
```

REVISION HISTORY:

```
    (1 ) Is overloaded by interface "IT_IS_NAN".
    (2 ) Now call C routine is_nan(x) for Linux platform (bmy, 6/13/02)
    (3 ) Eliminate IF statement in Linux section.  Also now trap NaN on
          the Sun/Sparc platform.  Rename cpp switch from DEC_COMPAQ to
          COMPAQ. (bmy, 3/23/03)
    (4 ) Added patches for IBM/AIX platform (gcc, bmy, 6/27/03)
    (5 ) Use LINUX error-trapping for INTEL_FC (bmy, 10/24/03)
    (6 ) Renamed SGI to SGI_MIPS, LINUX to LINUX_PGI, INTEL_FC to INTEL_IFC,
          and added LINUX_EFC. (bmy, 12/2/03)
    (7 ) Added LINUX_IFORT switch for Intel v8 and v9 compilers (bmy, 10/18/05)
    (8 ) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
    (9 ) Now use ISNAN for Linux/IFORT compiler (bmy, 8/14/07)
    (10) Remove support for SGI, COMPAQ compilers.  Add IBM_XLF switch.
          (bmy, 7/8/09)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.2 nan_dble

Function NAN_DBLE returns TRUE if a REAL*8 number is equal to the IEEE NaN (Not-a-Number) flag. Returns FALSE otherwise.

INTERFACE:

```
      FUNCTION NAN_DBLE( VALUE ) RESULT( IT_IS_A_NAN )
```

USES:

```
#     include "define.h"

#if   defined( IBM_AIX ) || defined( IBM_XLF )
      USE IEEE_ARITHMETIC
#endif
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: VALUE        ! Value to be tested for NaN
```

RETURN VALUE:

```
      LOGICAL            :: IT_IS_A_NAN  ! =T if VALUE is NaN; =F otherwise
```

REVISION HISTORY:

```
    (1 ) Is overloaded by interface "IT_IS_NAN".
    (2 ) Now call C routine is_nan(x) for Linux platform (bmy, 6/13/02)
    (3 ) Eliminate IF statement in Linux section.  Also now trap NaN on
          the Sun/Sparc platform.  Rename cpp switch from DEC_COMPAQ to
          COMPAQ. (bmy, 3/23/03)
    (4 ) Added patches for IBM/AIX platform (gcc, bmy, 6/27/03)
    (5 ) Use LINUX error-trapping for INTEL_FC (bmy, 10/24/03)
    (6 ) Renamed SGI to SGI_MIPS, LINUX to LINUX_PGI, INTEL_FC to INTEL_IFC,
          and added LINUX_EFC. (bmy, 12/2/03)
    (7 ) Added LINUX_IFORT switch for Intel v8 and v9 compilers (bmy, 10/18/05)
    (8 ) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
    (9 ) Now use ISNAN for Linux/IFORT compiler (bmy, 8/14/07)
    (10) Remove support for SGI, COMPAQ compilers.  Add IBM_XLF switch.
          (bmy, 7/8/09)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.3 finite_float

Function FINITE_FLOAT returns FALSE if a REAL*4 number is equal to the IEEE Infinity flag. Returns TRUE otherwise.

INTERFACE:

```
      FUNCTION FINITE_FLOAT( VALUE ) RESULT( IT_IS_A_FINITE )
```

USES:

```
#     include "define.h"

#if   defined( IBM_AIX ) || defined( IBM_XLF )
      USE IEEE_ARITHMETIC
#endif
```

INPUT PARAMETERS:

```
      REAL*4, INTENT(IN) :: VALUE           ! Value to be tested for infinity
```

RETURN VALUE:

```
      LOGICAL            :: IT_IS_A_FINITE  ! =T if VALUE is finite; =F else
```

REVISION HISTORY:

```
    (1 ) Is overloaded by interface "IT_IS_FINITE".
    (2 ) Now use correct values for bit masking (bmy, 11/15/01)
    (3 ) Eliminate IF statement in Linux section.  Also now trap Infinity on
          the Sun/Sparc platform.  Rename cpp switch from DEC_COMPAQ to
          COMPAQ. (bmy, 3/23/03)
    (4 ) Added patches for IBM/AIX platform (gcc, bmy, 6/27/03)
    (5 ) Bug fix: now use external C IS_FINITE for PGI/Linux (bmy, 9/29/03)
    (6 ) Use LINUX error-trapping for INTEL_FC (bmy, 10/24/03)
    (7 ) Renamed SGI to SGI_MIPS, LINUX to LINUX_PGI, INTEL_FC to INTEL_IFC,
          and added LINUX_EFC. (bmy, 12/2/03)
    (8 ) Added LINUX_IFORT switch for Intel v8 and v9 compilers (bmy, 10/18/05)
    (9 ) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
    (10) Now use FP_CLASS for IFORT compiler (bmy, 8/14/07)
    (11) Remove support for SGI, COMPAQ compilers.  Add IBM_XLF switch.
          (bmy, 7/8/09)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.4 finite_dble

Function FINITE_FLOAT returns FALSE if a REAL*8 number is equal to the IEEE Infinity flag. Returns TRUE otherwise.

INTERFACE:

```
      FUNCTION FINITE_DBLE( VALUE ) RESULT( IT_IS_A_FINITE )
```

USES:

```
#     include "define.h"

#if   defined( IBM_AIX ) || defined( IBM_XLF )
      USE IEEE_ARITHMETIC
#endif
```

INPUT PARAMETERS:

```
REAL*8, INTENT(IN) :: VALUE          ! Value to be tested for infinity
```

RETURN VALUE:

```
LOGICAL            :: IT_IS_A_FINITE  ! =T if VALUE is finite; =F else
```

REVISION HISTORY:

```
(1 ) Is overloaded by interface "IT_IS_FINITE".
(2 ) Now use correct values for bit masking (bmy, 11/15/01)
(3 ) Eliminate IF statement in Linux section.  Also now trap Infinity on
      the Sun/Sparc platform.  Rename cpp switch from DEC_COMPAQ to
      COMPAQ. (bmy, 3/23/03)
(4 ) Added patches for IBM/AIX platform (gcc, bmy, 6/27/03)
(5 ) Bug fix: now use external C IS_FINITE for PGI/Linux (bmy, 9/29/03)
(6 ) Use LINUX error-trapping for INTEL_FC (bmy, 10/24/03)
(7 ) Renamed SGI to SGI_MIPS, LINUX to LINUX_PGI, INTEL_FC to INTEL_IFC,
      and added LINUX_EFC. (bmy, 12/2/03)
(8 ) Added LINUX_IFORT switch for Intel v8 and v9 compilers (bmy, 10/18/05)
(9 ) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
(10) Now use FP_CLASS for IFORT compiler (bmy, 8/14/07)
(11) Remove support for SGI, COMPAQ compilers.  Add IBM_XLF switch.
      (bmy, 7/8/09)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.5 check_real_value

Subroutine CHECK_REAL_VALUE checks to make sure a REAL*4 value is not NaN or Infinity. This is a wrapper for the interfaces IT_IS_NAN and IT_IS_FINITE.

INTERFACE:

```
SUBROUTINE CHECK_REAL_VALUE( VALUE, LOCATION, VARNAME, MESSAGE )
```

INPUT PARAMETERS:

```
REAL*4,           INTENT(IN) :: VALUE        ! Value to be checked
CHARACTER(LEN=*), INTENT(IN) :: VARNAME      ! Name of variable
CHARACTER(LEN=*), INTENT(IN) :: MESSAGE      ! Short descriptive msg
INTEGER,          INTENT(IN) :: LOCATION(4)  ! (/ I, J, L, N /) indices
```

REVISION HISTORY:

```
13 Jun 2001 - R. Yantosca - Initial version
15 Oct 2002 - R. Yantosca - Now call GEOS_CHEM_STOP to shutdown safely
15 Oct 2002 - R. Yantosca - Updated comments, cosmetic changes
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.6 check_dble_value

Subroutine CHECK_DBLE_VALUE checks to make sure a REAL*4 value is not NaN or Infinity. This is a wrapper for the interfaces IT_IS_NAN and IT_IS_FINITE.

INTERFACE:

```
      SUBROUTINE CHECK_DBLE_VALUE( VALUE, LOCATION, VARNAME, MESSAGE )
```

INPUT PARAMETERS:

```
      REAL*8,            INTENT(IN) :: VALUE        ! Value to be checked
      CHARACTER(LEN=*), INTENT(IN) :: VARNAME      ! Name of variable
      CHARACTER(LEN=*), INTENT(IN) :: MESSAGE      ! Short descriptive msg
      INTEGER,           INTENT(IN) :: LOCATION(4)  ! (/ I, J, L, N /) indices
```

REVISION HISTORY:

```
   13 Jun 2001 - R. Yantosca - Initial version
   15 Oct 2002 - R. Yantosca - Now call GEOS_CHEM_STOP to shutdown safely
   15 Oct 2002 - R. Yantosca - Updated comments, cosmetic changes
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.7 error_stop

Subroutine ERROR_STOP is a wrapper for GEOS_CHEM_STOP. It prints an error message then calls GEOS_CHEM_STOP to free memory and quit.

INTERFACE:

```
      SUBROUTINE ERROR_STOP( MESSAGE, LOCATION )
```

INPUT PARAMETERS:

```
      CHARACTER(LEN=*), INTENT(IN) :: MESSAGE    ! Error msg to print
      CHARACTER(LEN=*), INTENT(IN) :: LOCATION   ! Where ERROR_STOP is called
```

REVISION HISTORY:

```
   15 Oct 2002 - R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.8 geos_chem_stop

Subroutine GEOS_CHEM_STOP calls CLEANUP to deallocate all module arrays and then stops the run.

INTERFACE:

```
      SUBROUTINE GEOS_CHEM_STOP
```

USES:

```
 #     include "define.h"
```

REVISION HISTORY:

```
    15 Oct 2002 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Now EXIT works for LINUX_IFC, LINUX_EFC,
                                so remove #if block.
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.9 alloc_err

Subroutine ALLOC_ERR prints an error message if there is not enough memory to allocate a particular allocatable array.

INTERFACE:

```
       SUBROUTINE ALLOC_ERR( ARRAYNAME, AS )
```

USES:

```
 #     include "define.h"
```

INPUT PARAMETERS:

```
       CHARACTER(LEN=*),  INTENT(IN) :: ARRAYNAME  ! Name of array
       INTEGER, OPTIONAL, INTENT(IN) :: AS         ! Error output from "STAT"
```

REVISION HISTORY:

```
    26 Jun 2000 - R. Yantosca - Initial version, split off from "ndxx_setup.f"
    15 Oct 2002 - R. Yantosca - Added to "error_mod.f"
    30 Nov 2005 - R. Yantosca - Call IFORT_ERRMSG for Intel Fortran compiler
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.10 debug_msg

Subroutine DEBUG_MSG prints a message to the stdout buffer and flushes. This is useful for determining the exact location where errors occur.

INTERFACE:

```
       SUBROUTINE DEBUG_MSG( MESSAGE )
```

USES:

```
 #     include "define.h"
```

INPUT PARAMETERS:

```
       CHARACTER(LEN=*), INTENT(IN) :: MESSAGE   ! Message to print
```

REVISION HISTORY:

```
   07 Jan 2002 - R. Yantosca - Initial version
   (1 ) Now just write the message and flush the buffer (bmy, 7/5/01)
   (2 ) Renamed from "paftop.f" to "debug_msg.f" (bmy, 1/7/02)
   (3 ) Bundled into "error_mod.f" (bmy, 11/22/02)
   (4 ) Now do not FLUSH the buffer for EFC compiler (bmy, 4/6/04)
   (5 ) Now add a little space for debug output (bmy, 4/10/06)
   (6 ) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.11 safe_div

Function SAFE_DIV performs ”safe division”, that is to prevent overflow, underlow, NaN, or infinity errors. An alternate value is returned if the division cannot be performed.

INTERFACE:

```
       FUNCTION SAFE_DIV( N,        D,
      &                   ALT_NAN,  ALT_OVER,
      &                   ALT_UNDER            ) RESULT( Q )
```

INPUT PARAMETERS:

```
       REAL*8,            INTENT(IN) :: N           ! Numerator
       REAL*8,            INTENT(IN) :: D           ! Denominator
       REAL*8,            INTENT(IN) :: ALT_NAN     ! Alternate value to be
                                                    !  returned if the division
                                                    !  is either NAN (0/0) or
                                                    !  leads to overflow (i.e.,
                                                    !  a too large number)
       REAL*8, OPTIONAL, INTENT(IN) :: ALT_OVER     ! Alternate value to be
                                                    !  returned if the division
                                                    !  leads to overflow (default
                                                    !  is ALT_NAN)
       REAL*8, OPTIONAL, INTENT(IN) :: ALT_UNDER    ! Alternate value to be
                                                    !  returned if the division
                                                    !  leads to underflow
                                                    !  (default is 0, but you
                                                    !  could use TINY() if you
                                                    !  want a non-zero result).
```

RETURN VALUE:

```
       REAL*8                        :: Q           ! Output from the division
```

REMARKS:

```
For more information, see the discussion on:
 http://groups.google.com/group/comp.lang.fortran/browse_thread/thread/8b367f44c419fa1d/
```

REVISION HISTORY:

```
26 Feb 2008 - P. Le Sager & R. Yantosca - Initial version
(1) Now can return different alternate values if NAN (that is 0/0),
    overflow (that is a too large number), or too small (that is greater
    than 0 but less than smallest possible number). Default value is
    zero in case of underflow (phs, 4/14/09)
(2) Some compiler options flush underflows to zero (-ftz for IFort).
     To think about it (phs, 4/14/09)
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.12 is_safe_div

Function IS_SAFE_DIV tests for "safe division", that is check if the division will overflow/underflow or hold NaN. .FALSE. is returned if the division cannot be performed. (phs, 6/11/08)

INTERFACE:

```
   FUNCTION IS_SAFE_DIV( N, D, R4 ) RESULT( F )
```

INPUT PARAMETERS:

```
   REAL*8,  INTENT(IN)           :: N    ! Numerator
   REAL*8,  INTENT(IN)           :: D    ! Denominator
   LOGICAL, INTENT(IN), OPTIONAL :: R4   ! Logical flag to use the limits
                                         !  of REAL*4 to define underflow
                                         !  or overflow.  Extra defensive.
```

OUTPUT PARAMETERS:

```
   LOGICAL                       :: F    ! =F if division isn't allowed
                                         ! =T otherwise
```

REMARKS:

```
UnderFlow, OverFlow and NaN are tested for. If you need to
differentiate between the three, use the SAFE_DIV (phs, 4/14/09)
```

REVISION HISTORY:

```
11 Jun 2008 - P. Le Sager - Initial version
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.4.13 safe_exp

Function SAFE_EXP performs a "safe exponential", that is to prevent overflow, underlow, NaN, or infinity errors when taking the value EXP(x). An alternate value is returned if the exponential cannot be performed.

INTERFACE:

```
      FUNCTION SAFE_EXP( X, ALT ) RESULT( VALUE )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: X      ! Argument of EXP
      REAL*8, INTENT(IN) :: ALT    ! Alternate value to be returned
```

RETURN VALUE:

```
      REAL*8             :: VALUE  ! Output from the exponential
```

REVISION HISTORY:

```
    04 Jan 2010 - R. Yantosca - Initial version
```

1.4.14 is_safe_exp

Function IS_SAFE_EXP returns TRUE if it is safe to take the value EXP(x) without encountering a floating point exception. FALSE is returned if the exponential cannot be performed.

INTERFACE:

```
      FUNCTION IS_SAFE_EXP( X ) RESULT( F )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: X    ! Argument to the exponential function
```

OUTPUT PARAMETERS:

```
      LOGICAL            :: F    ! =F if exponential isn't allowed
                                 ! =T otherwise
```

REMARKS:

```
    Empirical testing has revealed that -600 < X < 600 will not result in
    a floating-point exception on Sun and IFORT compilers.  This is good
    enough for most purposes.
```

REVISION HISTORY:

```
    04 Jan 2010 - R. Yantosca - Initial version
```

1.4.15 safe_log

Function SAFE_LOG performs a "safe natural logarithm", that is to prevent overflow, underlow, NaN, or infinity errors when taking the value LOG(x). An alternate value is returned if the logarithm cannot be performed.

INTERFACE:

```
      FUNCTION SAFE_LOG( X, ALT ) RESULT( VALUE )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: X      ! Argument of LOG
      REAL*8, INTENT(IN) :: ALT    ! Alternate value to be returned
```

RETURN VALUE:

```
      REAL*8             :: VALUE  ! Output from the natural logarithm
```

REVISION HISTORY:

```
    04 Jan 2010 - R. Yantosca - Initial version
```

1.4.16 safe_log10

Function SAFE_LOG10 performs a "safe log10", that is to prevent overflow, underlow, NaN, or infinity errors when taking the value LOG10(x). An alternate value is returned if the logarithm cannot be performed.

INTERFACE:

```
      FUNCTION SAFE_LOG10( X, ALT ) RESULT( VALUE )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: X      ! Argument of LOG10
      REAL*8, INTENT(IN) :: ALT    ! Alternate value to be returned
```

RETURN VALUE:

```
      REAL*8             :: VALUE  ! Output from the natural logarithm
```

REVISION HISTORY:

```
    04 Jan 2010 - R. Yantosca - Initial version
```

1.5 Fortran: Module Interface file_mod.f

Module FILE_MOD contains file unit numbers, as well as file I/O routines for GEOS-Chem. FILE_MOD keeps all of the I/O unit numbers in a single location for convenient access.

INTERFACE:

```
MODULE FILE_MOD
```

USES:

```
IMPLICIT NONE
PRIVATE
```

DEFINED PARAMETERS:

```
! Logical file unit numbers for ...
INTEGER, PUBLIC, PARAMETER :: IU_RST     = 1   ! Tracer restart file
INTEGER, PUBLIC, PARAMETER :: IU_CHEMDAT = 7   ! "chem.dat"
INTEGER, PUBLIC, PARAMETER :: IU_FASTJ   = 8   ! FAST-J input files
INTEGER, PUBLIC, PARAMETER :: IU_GEOS    = 10  ! "input.geos"
INTEGER, PUBLIC, PARAMETER :: IU_BPCH    = 11  ! "ctm.bpch"
INTEGER, PUBLIC, PARAMETER :: IU_ND20    = 12  ! "rate.YYYYMMDD"
INTEGER, PUBLIC, PARAMETER :: IU_ND48    = 13  ! ND48 output
INTEGER, PUBLIC, PARAMETER :: IU_ND49    = 14  ! "tsYYYYMMDD.bpch"
INTEGER, PUBLIC, PARAMETER :: IU_ND50    = 15  ! "ts24h.bpch"
INTEGER, PUBLIC, PARAMETER :: IU_ND51    = 16  ! "ts10_12am.bpch" etc.
INTEGER, PUBLIC, PARAMETER :: IU_ND51b   = 23  ! for ND51b diagnostic
INTEGER, PUBLIC, PARAMETER :: IU_ND52    = 17  ! ND52 output (NRT only)
INTEGER, PUBLIC, PARAMETER :: IU_PLANE   = 18  ! "plane.log"
INTEGER, PUBLIC, PARAMETER :: IU_BC      = 19  ! TPCORE BC files
INTEGER, PUBLIC, PARAMETER :: IU_BC_NA   = 20  ! TPCORE BC files: NA grid
INTEGER, PUBLIC, PARAMETER :: IU_BC_EU   = 21  ! TPCORE BC files: EU grid
INTEGER, PUBLIC, PARAMETER :: IU_BC_CH   = 22  ! TPCORE BC files: CH grid
INTEGER, PUBLIC, PARAMETER :: IU_FILE    = 65  ! Generic file
INTEGER, PUBLIC, PARAMETER :: IU_TP      = 69  ! "YYYYMMDD.tropp.*"
INTEGER, PUBLIC, PARAMETER :: IU_PH      = 70  ! "YYYYMMDD.phis.*"
INTEGER, PUBLIC, PARAMETER :: IU_I6      = 71  ! "YYYYMMDD.i6.*"
INTEGER, PUBLIC, PARAMETER :: IU_A6      = 72  ! "YYYYMMDD.a6.*"
INTEGER, PUBLIC, PARAMETER :: IU_A3      = 73  ! "YYYYMMDD.a3.*"
INTEGER, PUBLIC, PARAMETER :: IU_A1      = 74  ! "YYYYMMDD.a1.*"
INTEGER, PUBLIC, PARAMETER :: IU_GWET    = 75  ! "YYYYMMDD.gwet.*"
INTEGER, PUBLIC, PARAMETER :: IU_XT      = 76  ! "YYYYMMDD.xtra.*"
INTEGER, PUBLIC, PARAMETER :: IU_CN      = 77  ! "YYYYMMDD.cn.*"
INTEGER, PUBLIC, PARAMETER :: IU_SMV2LOG = 93  ! "smv2.log"
INTEGER, PUBLIC, PARAMETER :: IU_DEBUG   = 98  ! Reserved for debugging
INTEGER, PUBLIC, PARAMETER :: IU_OAP     = 99  ! soaprod.YYYYMMDDhh
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC  :: CLOSE_FILES
PUBLIC  :: FILE_EXISTS
PUBLIC  :: IOERROR

INTERFACE FILE_EXISTS
   MODULE PROCEDURE FILE_EX_C
   MODULE PROCEDURE FILE_EX_I
END INTERFACE
```

PRIVATE MEMBER FUNCTIONS:

```
PRIVATE :: FILE_EX_C
PRIVATE :: FILE_EX_I
```

REVISION HISTORY:

```
(1 ) Moved "ioerror.f" into this module. (bmy, 7/1/02)
(2 ) Now references "error_mod.f" (bmy, 10/15/02)
(3 ) Renamed cpp switch from DEC_COMPAQ to COMPAQ.  Also added code to
      trap I/O errors on SUN/Sparc platform. (bmy, 3/23/03)
(4 ) Now added IU_BC for nested boundary conditions as unit 18
      (bmy, 3/27/03)
(5 ) Renamed IU_CTMCHEM to IU_SMV2LOG (bmy, 4/21/03)
(6 ) Now print out I/O errors for IBM and INTEL_FC compilers (bmy, 11/6/03)
(7 ) Changed the name of some cpp switches in "define.h" (bmy, 12/2/03)
(8 ) Renumbered the order of the files.  Also removed IU_INPTR and
      IU_INPUT since they are now obsolete. (bmy, 7/20/04)
(9 ) Added overloaded routines FILE_EX_C and FILE_EX_I (bmy, 3/23/05)
(10) Added LINUX_IFORT switch for Intel v8 & v9 compilers (bmy, 10/18/05)
(11) Added IU_XT for GEOS3 XTRA met fields files for MEGAN (tmf, 10/20/05)
(12) Extra modification for Intel v9 compiler (bmy, 11/2/05)
(13) Now print IFORT error messages (bmy, 11/30/05)
(14) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
(15) Remove support for SGI & COMPAQ compilers (bmy, 7/8/09)
20 Nov 2009 - R. Yantosca - Added ProTeX headers
18 Dec 2009 - Aaron van D - Added file units IU_BC_NA, IU_BC_EU, IU_BC_CH
15 Mar 2010 - D. Henze    - Add IU_OAP for SOA restart file.
19 Aug 2010 - R. Yantosca - Added IU_CN and IU_A1 parameters for MERRA
19 Aug 2010 - R. Yantosca - Remove IU_KZZ
```

1.5.1 ioerror

Subroutine IOERROR prints out I/O error messages. The error number, file unit, location, and a brief description will be printed, and program execution will be halted. (bmy, 5/28/99, 7/4/09)

INTERFACE:

```
SUBROUTINE IOERROR( ERROR_NUM, UNIT, LOCATION )
```

USES:

```
      USE ERROR_MOD, ONLY : GEOS_CHEM_STOP

#     include "define.h"                      ! C-preprocessor switches
```

INPUT PARAMETERS:

```
      INTEGER,          INTENT(IN) :: ERROR_NUM  ! I/O error from IOSTAT
      INTEGER,          INTENT(IN) :: UNIT       ! Logical unit # for file
      CHARACTER(LEN=*), INTENT(IN) :: LOCATION   ! Descriptive message
```

REVISION HISTORY:

```
  (1 ) Now flush the standard output buffer before stopping.
        Also updated comments. (bmy, 2/7/00)
  (2 ) Changed ROUTINE_NAME to LOCATION.  Now also use C-library routines
        gerror and strerror() to get the error string corresponding to
        ERROR_NUM.  For SGI platform, also print the command string that
        will call the SGI "explain" command, which will yield additional
        information about the error.  Updated comments, cosmetic changes.
        Now also reference "define.h". (bmy, 3/21/02)
  (3 ) Moved into "file_mod.f".  Now reference GEOS_CHEM_STOP from module
        "error_mod.f".  Updated comments, cosmetic changes. (bmy, 10/15/02)
  (4 ) Renamed cpp switch from DEC_COMPAQ to COMPAQ.  Also added code to
        display I/O errors on SUN platform. (bmy, 3/23/03)
  (5 ) Now call GERROR for IBM and INTEL_FC compilers (bmy, 11/6/03)
  (6 ) Renamed SGI to SGI_MIPS, LINUX to LINUX_PGI, INTEL_FC to INTEL_IFC,
        and added LINUX_EFC. (bmy, 12/2/03)
  (7 ) Now don't flush the buffer for LINUX_EFC (bmy, 4/23/04)
  (8 ) Modifications for Linux/IFORT Intel v9 compiler (bmy, 11/2/05)
  (9 ) Now call IFORT_ERRMSG to get the IFORT error messages (bmy, 11/30/05)
  (10) Remove support for LINUX_IFC & LINUX_EFC compilers (bmy, 8/4/06)
  (10) Remove support for SGI & COMPAQ compilers.  Add IBM_XLF switch.
        (bmy, 7/8/09)
  20 Nov 2009 - R. Yantosca - Removed commented-out code for SGI, COMPAQ
  20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.5.2 file_ex_c

Function FILE_EX_C returns TRUE if FILENAME exists or FALSE otherwise. This is handled in a platform-independent way. The argument is of CHARACTER type.

INTERFACE:

```
      FUNCTION FILE_EX_C( FILENAME ) RESULT( IT_EXISTS )
```

USES:

```
#     include "define.h"
```

INPUT PARAMETERS:

```
       CHARACTER(LEN=*), INTENT(IN) :: FILENAME   ! Name of file or dir to test
```

RETURN VALUE:

```
       LOGICAL                      :: IT_EXISTS  ! =T if the file/dir exists
```

REMARKS:

```
    This routine is overloaded by public interface FILE_EXISTS.
```

REVISION HISTORY:

```
    23 Mar 2005 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Updated for LINUX/IFORT Intel v9 compiler
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.5.3 file_ex_i

Function FILE_EX_I returns TRUE if FILENAME exists or FALSE otherwise. This is handled in a platform-independent way. The argument is of INTEGER type.

INTERFACE:

```
       FUNCTION FILE_EX_I( IUNIT ) RESULT( IT_EXISTS )
```

USES:

```
 #     include "define.h"
```

INPUT PARAMETERS:

```
       ! Arguments
       INTEGER, INTENT(IN) :: IUNIT      ! LUN of file to be tested
```

RETURN VALUE:

```
       LOGICAL             :: IT_EXISTS  ! =T if the file/dir exists
```

REMARKS:

```
    This routine is overloaded by public interface FILE_EXISTS.
```

REVISION HISTORY:

```
    23 Mar 2005 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.5.4 close_files

Subroutine CLOSE_FILES closes files used by GEOS-Chem. This should be called only from the end of the "main.f" program.

INTERFACE:

```
      SUBROUTINE CLOSE_FILES
```

REVISION HISTORY:

```
    04 Mar 1998 - R. Yantosca - Initial version
    27 Jun 2002 - R. Yantosca - Moved into "file_mod.f"
    27 Mar 2003 - R. Yantosca - Also close IU_BC
    20 Jul 2004 - R. Yantosca - Removed obsolete IU_INPUT and IU_INPTR.
    20 Jul 2004 - R. Yantosca - Also renamed IU_TS to IU_ND48.
    20 Oct 2005 - R. Yantosca - Also close IU_XT.
    20 Nov 2009 - R. Yantosca - Added ProTeX header
    18 Dec 2009 - Aaron van D - Now close files IU_BC_NA, IU_BC_EU, IU_BC_CH
    19 Aug 2010 - R. Yantosca - Remove IU_KZZ
    19 Aug 2010 - R. Yantosca - Now close IU_A1
```

1.6 Fortran: Module Interface grid_mod.f

Module GRID_MOD contains variables and routines which are used to specify the parameters of a GEOS-CHEM horizontal grid.

INTERFACE:

```
      MODULE GRID_MOD
```

USES:

```
      IMPLICIT NONE
      PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
      PUBLIC  :: CLEANUP_GRID
      PUBLIC  :: COMPUTE_GRID
      PUBLIC  :: GET_AREA_M2
      PUBLIC  :: GET_AREA_CM2
      PUBLIC  :: GET_BOUNDING_BOX
      PUBLIC  :: GET_XEDGE
      PUBLIC  :: GET_XMID
      PUBLIC  :: GET_XOFFSET
      PUBLIC  :: GET_YOFFSET
      PUBLIC  :: GET_YEDGE
      PUBLIC  :: GET_YEDGE_R
```

```
       PUBLIC  :: GET_YMID
       PUBLIC  :: GET_YMID_R
       PUBLIC  :: GET_YMID_R_W
       PUBLIC  :: SET_XOFFSET
       PUBLIC  :: SET_YOFFSET
       PUBLIC  :: ITS_A_NESTED_GRID
```

PRIVATE MEMBER FUNCTIONS:

```
       PRIVATE :: INIT_GRID
```

REVISION HISTORY:

```
   11 Mar 2003 - R. Yantosca - Initial version
   (1 ) Fixed typos in "grid_mod.f" (bmy, 4/28/03)
   (2 ) Added routine GET_BOUNDING_BOX.  Now define 1x125 grid. (bmy, 12/1/04)
   (3 ) Modified for GCAP 4x5 horizontal grid (swu, bmy, 5/24/05)
   (4 ) Added comments re: surface area derivation (bmy, 4/20/06)
   (5 ) Modifications for GEOS-5 nested grids (yxw, dan, bmy, 11/6/08)
   20 Nov 2009 - R. Yantosca - Added ProTeX headers
```

1.6.1 compute_grid

Subroutine COMPUTE_GRID initializes the longitude, latitude and surface area arrays.

INTERFACE:

```
       SUBROUTINE COMPUTE_GRID
```

USES:

```
 #     include "CMN_SIZE"  ! Size parameters
 #     include "CMN_GCTM"  ! Physical constants
```

REVISION HISTORY:

```
   11 Mar 2003 - R. Yantosca - Initial version
   (1 ) Added fancy output (bmy, 4/26/04)
   (2 ) Suppress some output lines (bmy, 7/20/04)
   (3 ) Now also support 1 x 1.25 grid (bmy, 12/1/04)
   (4 ) Now modified for GCAP 4x5 horizontal grid (swu, bmy, 5/24/05)
   (5 ) Added comments re: surface area derivation (bmy, 4/20/06)
   (6 ) Compute YMID, YEDGE for 0.5x0.666 nested grids (yxw, dan, bmy, 11/6/08)
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.2 set_xoffset

Function SET_XOFFSET initializes the nested-grid longitude offset variable I0.

INTERFACE:

```
       SUBROUTINE SET_XOFFSET( X_OFFSET )
```

INPUT PARAMETERS:

```
       INTEGER, INTENT(IN) :: X_OFFSET  ! Value to assign to I0
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.3 set_yoffset

Function SET_YOFFSET initializes the nested-grid latitude offset variable J0.

INTERFACE:

```
       SUBROUTINE SET_YOFFSET( Y_OFFSET )
```

INPUT PARAMETERS:

```
       INTEGER, INTENT(IN) :: Y_OFFSET  ! Value to assign to J0
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.4 get_xoffset

Function GET_XOFFSET returns the nested-grid longitude offset to the calling program. (bmy, 3/11/03)

INTERFACE:

```
       FUNCTION GET_XOFFSET( GLOBAL ) RESULT( X_OFFSET )
```

INPUT PARAMETERS:

```
       ! If GLOBAL is passed, then return the actual window offset.
       ! This is necessary for certain instances (e.g. diagnostics)
       LOGICAL, INTENT(IN), OPTIONAL :: GLOBAL
```

RETURN VALUE:

```
       INTEGER                       :: X_OFFSET
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.5 get_xoffset

Function GET_XOFFSET returns the nested-grid longitude offset to the calling program. (bmy, 3/11/03)

INTERFACE:

```
      FUNCTION GET_YOFFSET( GLOBAL ) RESULT( Y_OFFSET )
```

INPUT PARAMETERS:

```
      ! If GLOBAL is passed, then return the actual window offset.
      ! This is necessary for certain instances (e.g. diagnostics)
      LOGICAL, INTENT(IN), OPTIONAL :: GLOBAL
```

RETURN VALUE:

```
      INTEGER                       :: Y_OFFSET
```

REVISION HISTORY:

```
   11 Mar 2003 - R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.6 get_xmid

Function GET_XMID returns the longitude in degrees at the center of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_XMID( I ) RESULT( X )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: I  ! Longitude index
```

RETURN VALUE:

```
      REAL*8              :: X  ! Corresponding lon value @ grid box ctr
```

REVISION HISTORY:

```
   11 Mar 2003 - R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.7 get_xedge

Function GET_XEDGE returns the longitude in degrees at the western edge of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_XEDGE( I ) RESULT( X )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: I  ! Longitude index
```

RETURN VALUE:

```
      REAL*8              :: X  ! Corresponding lon value @ W edge of grid box
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.8 get_ymid

Function GET_YMID returns the latitude in degrees at the center of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_YMID( J ) RESULT( Y )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: Y  ! Latitude value at @ grid box ctr [degrees]
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.9 get_yedge

Function GET_YEDGE returns the latitude in degrees at the southern edge of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_YEDGE( J ) RESULT( Y )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: Y  ! Latitude value @ S edge of grid box [degrees]
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.10 get_ymid

Function GET_YMID_R returns the latitude in radians at the center of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_YMID_R( J ) RESULT( Y )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: Y  ! Latitude value at @ grid box ctr [radians]
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.11 get_ymid

Function GET_YMID_R_W returns the latitude in radians at the center of a GEOS-Chem grid box for the GEOS-5 nested grid.

INTERFACE:

```
      FUNCTION GET_YMID_R_W( J ) RESULT( Y )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: Y  ! Latitude value at @ grid box ctr [radians]
```

REVISION HISTORY:

```
    06 Nov 2008 - D. Chen & R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.12 get_yedge_r

Function GET_YEDGE_R returns the latitude in radians at the southern edge of a GEOS-Chem grid box. Works for nested-grids too.

INTERFACE:

```
      FUNCTION GET_YEDGE_R( J ) RESULT( Y )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: Y  ! Latitude value @ S edge of grid box [radians]
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.13 get_area_m2

Function GET_AREA_M2 returns the surface area [m2] of a GEOS-Chem grid box. Works for nested grids too.

INTERFACE:

```
      FUNCTION GET_AREA_M2( J ) RESULT( A )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: A  ! Grid box surface area [m2]
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.14 get_area_cm2

Function GET_AREA_CM2 returns the surface area [cm2] of a GEOS-Chem grid box. Works for nested grids too.

INTERFACE:

```
      FUNCTION GET_AREA_CM2( J ) RESULT( A )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: J  ! Latitude index
```

RETURN VALUE:

```
      REAL*8              :: A  ! Grid box surface area [cm2]
```

REVISION HISTORY:

```
   11 Mar 2003 - R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.15 get_bounding_box

Subroutine GET_BOUNDING_BOX returns the indices which specify the lower left (LL) and upper right (UR) corners of a rectangular region, given the corresponding longitude and latitude values.

INTERFACE:

```
      SUBROUTINE GET_BOUNDING_BOX( COORDS, INDICES )
```

USES:

```
      USE ERROR_MOD, ONLY : ERROR_STOP

#     include "CMN_SIZE"    ! Size parameters
```

INPUT PARAMETERS:

```
      REAL*8,  INTENT(IN)  :: COORDS(4)   ! (/LON_LL, LAT_LL, LON_UR, LAT_UR/)
```

INPUT/OUTPUT PARAMETERS:

```
      INTEGER, INTENT(OUT) :: INDICES(4)  ! (/I_LL, J_LL, I_UR, J_UR/)
```

REVISION HISTORY:

```
   01 Dec 2004 - R. Yantosca - Initial version
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.16 its_a_nested_grid

Function GET_AREA_CM2 returns the surface area [cm2] of a GEOS-Chem grid box. Works for nested grids too.

INTERFACE:

```
      FUNCTION ITS_A_NESTED_GRID() RESULT( IT_IS_NESTED )
```

RETURN VALUE:

```
       LOGICAL :: IT_IS_NESTED   ! =T if it's a nested grid; =F otherwise
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.17 init_grid

Subroutine INIT_GRID initializes variables and allocates module arrays.

INTERFACE:

```
       SUBROUTINE INIT_GRID
```

USES:

```
       USE ERROR_MOD, ONLY : ALLOC_ERR

 #     include "CMN_SIZE"
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    (1 ) Fixed typos that caused AREA_CM2_G and AREA_CM2 to be initialized
          before they were allocated. (bmy, 4/28/03)
    (2 ) Now define IIGLOB & JJGLOB for 1 x 1.25 grid (bmy, 12/1/04)
    (3 ) Modified for GCAP 4x5 horizontal grid (swu, bmy, 5/24/05)
    (4 ) Modifications for 0.5 x 0.666 nested grids (dan, bmy, 11/6/08)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.6.18 cleanup_grid

Subroutine CLEANUP_GRID deallocates all module arrays.

INTERFACE:

```
       SUBROUTINE CLEANUP_GRID
```

REVISION HISTORY:

```
    11 Mar 2003 - R. Yantosca - Initial version
    20 Nov 2009 - D. Chen     - Now also deallocate YMID_R_W
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.7 Fortran: Module Interface hdf_mod

Module HDF_MOD contains routines to write data to HDF5 files.

INTERFACE:

```
MODULE HDF_MOD
```

USES:

```
IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
PUBLIC :: CLEANUP_HDF
PUBLIC :: CLOSE_HDF
PUBLIC :: INIT_HDF
PUBLIC :: OPEN_HDF
PUBLIC :: WRITE_HDF
```

PUBLIC DATA MEMBERS:

```
PUBLIC :: HDFCATEGORY
PUBLIC :: HDFDESCRIPT
PUBLIC :: HDFNAME
PUBLIC :: HDFFNAME
PUBLIC :: HDFMOLC
PUBLIC :: HDFMWT
PUBLIC :: HDFSCALE
PUBLIC :: HDFUNIT

CHARACTER(LEN=40), ALLOCATABLE :: HDFCATEGORY(:)
CHARACTER(LEN=40), ALLOCATABLE :: HDFDESCRIPT(:)
INTEGER                        :: MAXDIAG
INTEGER                        :: MAXTRACER
INTEGER                        :: MAXCAT
INTEGER,           ALLOCATABLE :: HDFMOLC(:,:)
REAL*4,            ALLOCATABLE :: HDFMWT(:,:)
REAL*4,            ALLOCATABLE :: HDFSCALE(:,:)
CHARACTER(LEN=40), ALLOCATABLE :: HDFNAME(:,:)
CHARACTER(LEN=40), ALLOCATABLE :: HDFFNAME(:,:)
CHARACTER(LEN=40), ALLOCATABLE :: HDFUNIT(:,:)
```

REMARKS:

```
If you have the HDF5 library installed on your system, then you can
compile GEOS-Chem with the option:

   make HDF5=yes
```

```
    which will activate the HDF5-specific code in this module to enable file
    I/O to HDF5 format.  You must also specify the HDF5 include and library
    paths in the Makefile_header.mk.

    The default is not to activate the HDF5-specific code.
```

REVISION HISTORY:

```
    19 Nov 2009 - A. van Donkelaar - Initial Version
    21 Dec 2009 - R. Yantosca      - Modified to block out HDF5-specific code
                                     so that users w/o HDF5 library can still
                                     compile & run GEOS-Chem
    21 Dec 2009 - R. Yantosca      - Updated comments
```

1.7.1 open_hdf

Subroutine OPEN_HDF creates and opens an hdf file for output.

INTERFACE:

```
 #if   defined( USE_HDF5 )
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       !%%%    NOTE: Subroutine is used only when USE_HDF5 is defined!    %%%
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       SUBROUTINE OPEN_HDF( IU_HDF, FILENAME, IMAX, IMIN,
      &                     JMAX,   JMIN,     NI,   NJ    )
```

USES:

```
       USE HDF5
       USE GRID_MOD, ONLY : GET_XMID
       USE GRID_MOD, ONLY : GET_YMID

 #     include "CMN_SIZE"    ! Size parameters
```

INPUT PARAMETERS:

```
       CHARACTER(LEN=*), INTENT(IN)    :: FILENAME   ! File name to open
       INTEGER,          INTENT(IN)    :: IMIN       ! Min longitude index
       INTEGER,          INTENT(IN)    :: IMAX       ! Max longitude index
       INTEGER,          INTENT(IN)    :: JMIN       ! Min latitude index
       INTEGER,          INTENT(IN)    :: JMAX       ! Max latitude index
       INTEGER,          INTENT(IN)    :: NI         ! # of longitudes
       INTEGER,          INTENT(IN)    :: NJ         ! # of latitudes
```

INPUT/OUTPUT PARAMETERS:

```
       INTEGER(HID_T),   INTENT(INOUT) :: IU_HDF     ! HDF5 file identifier
```

REVISION HISTORY:

```
    Nov 20 2009 - A. van Donkelaar - Initial Version
    21 Dec 2009 - R. Yantosca      - Modified to block out HDF5-specific code
                                     so that users w/o HDF5 library can still
                                     compile & run GEOS-Chem
    21 Dec 2009 - R. Yantosca      - Updated comments
```

1.7.2 close_hdf

Subroutine CLOSE_HDF closes an HDF5 file that is already open.

INTERFACE:

```
 #if   defined( USE_HDF5 )
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       !%%%    NOTE: Subroutine is used only when USE_HDF5 is defined!    %%%
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       SUBROUTINE CLOSE_HDF( IU_HDF )
```

USES:

```
       USE HDF5
```

INPUT/OUTPUT PARAMETERS:

```
       INTEGER(HID_T), INTENT(INOUT) :: IU_HDF  ! HDF5 File identifier
```

REVISION HISTORY:

```
    20 Nov 2009 - A. van Donkelaar - Initial Version
    21 Dec 2009 - R. Yantosca      - Modified to block out HDF5-specific code
                                     so that users w/o HDF5 library can still
                                     compile & run GEOS-Chem
    21 Dec 2009 - R. Yantosca      - Updated comments
```

1.7.3 write_hdf

Subroutine WRITE_HDF writes data to an open HDF5 file.

INTERFACE:

```
 #if   defined( USE_HDF5 )
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       !%%%    NOTE: Subroutine is used only when USE_HDF5 is defined!    %%%
       !%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
       SUBROUTINE WRITE_HDF( IU_HDF,  N,      NDCATEGORY, NDTRACER,
      &                      NDUNIT,  TAU0,   TAU1,       RESERVED,
      &                      NI,      NJ,     NL,         IFIRST,
      &                      JFIRST,  LFIRST, ARRAY )
```

USES:

```
USE HDF5
```

INPUT PARAMETERS:

```
INTEGER,           INTENT(IN) :: IU_HDF           ! HDF file unit #
INTEGER,           INTENT(IN) :: N                ! Actual tracer #
CHARACTER(LEN=40), INTENT(IN) :: NDCATEGORY       ! Diagnostic category
INTEGER,           INTENT(IN) :: NDTRACER         ! Tracer # for file
CHARACTER(LEN=40), INTENT(IN) :: NDUNIT           ! Units of data
REAL*8,            INTENT(IN) :: TAU0             ! TAU at start & end
REAL*8,            INTENT(IN) :: TAU1             !  of diag interval
CHARACTER(LEN=40), INTENT(IN) :: RESERVED         ! Descriptive string
INTEGER,           INTENT(IN) :: NI               ! # of longitudes
INTEGER,           INTENT(IN) :: NJ               ! # of latitudes
INTEGER,           INTENT(IN) :: NL               ! # of levels
INTEGER,           INTENT(IN) :: IFIRST           ! Index of 1st lon
INTEGER,           INTENT(IN) :: JFIRST           ! Index of 1st lat
INTEGER,           INTENT(IN) :: LFIRST           ! Index of 1st lev
REAL*4,            INTENT(IN) :: ARRAY(NI,NJ,NL)  ! Data array
```

REVISION HISTORY:

```
20 Nov 2009 - A. van Donkelaar - Initial Version
21 Dec 2009 - R. Yantosca      - Modified to block out HDF5-specific code
                                 so that users w/o HDF5 library can still
                                 compile & run GEOS-Chem
21 Dec 2009 - R. Yantosca      - Updated comments
```

1.7.4 init_hdf

Subroutine INIT_HDF allocates all module variables.

INTERFACE:

```
SUBROUTINE INIT_HDF( GMMAXCAT, GMMAXTRACER, GMMAXDIAG )
```

USES:

```
USE ERROR_MOD, ONLY : ALLOC_ERR
```

INPUT PARAMETERS:

```
INTEGER, INTENT(IN) :: GMMAXCAT
INTEGER, INTENT(IN) :: GMMAXTRACER
INTEGER, INTENT(IN) :: GMMAXDIAG
```

REVISION HISTORY:

```
    23 Nov 2009 - A. van Donkelaar - Initial Version
    21 Dec 2009 - R. Yantosca      - Modified to block out HDF5-specific code
                                     so that users w/o HDF5 library can still
                                     compile & run GEOS-Chem
    21 Dec 2009 - R. Yantosca      - Updated comments
```

1.7.5 cleanup_hdf

Subroutine CLEANUP_HDF deallocates all module variables.

INTERFACE:

```
      SUBROUTINE CLEANUP_HDF
```

REVISION HISTORY:

```
    21 Dec 2009 - R. Yantosca - Initial version
```

1.7.6 ifort_errmsg

Function IFORT_ERRMSG returns an error message string that corresponds to an I/O error number obtained via the IOSTAT or STAT specifiers. (This is specifically for the Intel Fortran compiler.)

INTERFACE:

```
      FUNCTION IFORT_ERRMSG( ERROR_NUM ) RESULT( MSG )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: ERROR_NUM   ! Error condition from IOSTAT
```

RETURN VALUE:

```
      CHARACTER(LEN=255)  :: MSG         ! Descriptive error message
```

REVISION HISTORY:

```
    30 Nov 2005 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.8 Fortran: Module Interface julday_mod.f

Module JULDAY_MOD contains routines used to convert from month/day/year to Astronomical Julian Date and back again.

INTERFACE:

```
      MODULE JULDAY_MOD
```

USES:

```
      IMPLICIT NONE
      PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
      PUBLIC  :: JULDAY
      PUBLIC  :: CALDATE
```

PRIVATE MEMBER FUNCTIONS:

```
      PRIVATE :: MINT
```

REVISION HISTORY:

```
  (1 ) Moved JULDAY, MINT, CALDATE here from "bpch2_mod.f" (bmy, 11/20/01)
  (2 ) Bug fix: now compute NHMS correctly.  Also use REAL*4 variables to
        avoid roundoff errors. (bmy, 11/26/01)
  (3 ) Updated comments (bmy, 5/28/02)
  (4 ) Renamed arguments for clarity (bmy, 6/26/02)
  20 Nov 2009 - R. Yantosca - Added ProTeX Headers
```

1.8.1 julday

Function JULDAY returns the astronomical Julian day.

INTERFACE:

```
      FUNCTION JULDAY( YYYY, MM, DD ) RESULT( JULIANDAY )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: YYYY        ! Year (must be in 4-digit format!)
      INTEGER, INTENT(IN) :: MM          ! Month (1-12)
      REAL*8,  INTENT(IN) :: DD          ! Day of month (may be fractional!)
```

RETURN VALUE:

```
      REAL*8              :: JULIANDAY   ! Astronomical Julian Date
```

REMARKS:

```
  (1) Algorithm taken from "Practical Astronomy With Your Calculator",
       Third Edition, by Peter Duffett-Smith, Cambridge UP, 1992.
  (2) Requires the external function MINT.F.
  (3) JulDay will compute the correct Julian day for any BC or AD date.
  (4) For BC dates, subtract 1 from the year and append a minus sign.
       For example, 1 BC is 0, 2 BC is -1, etc.  This is necessary for
       the algorithm.
```

REVISION HISTORY:

```
    26 Nov 2001 - R. Yantosca - Initial version
    Changed YEAR to YYYY, MONTH to MM, and DAY to DD for documentation
     purposes. (bmy, 6/26/02)
    20 Nov 2009 - R. Yantosca - Added ProTeX headers
```

1.8.2 mint

Function MINT is the modified integer function.

INTERFACE:

```
       FUNCTION MINT( X ) RESULT ( VALUE )
```

INPUT PARAMETERS:

```
       REAL*8, INTENT(IN) :: X
```

RETURN VALUE:

```
       REAL*8             :: VALUE
```

REMARKS:

```
    The modified integer function is defined as follows:
              { -INT( ABS( X ) )   for X < 0
       MINT = {
              {  INT( ABS( X ) )   for X >= 0
```

REVISION HISTORY:

```
    20 Nov 2001 - R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX headers
```

1.8.3 caldate

Subroutine CALDATE converts an astronomical Julian day to the YYYYMMDD and HHMMSS format.

INTERFACE:

```
       SUBROUTINE CALDATE( JULIANDAY, YYYYMMDD, HHMMSS )
```

INPUT PARAMETERS:

```
       ! Arguments
       REAL*8,  INTENT(IN)  :: JULIANDAY  ! Astronomical Julian Date
```

OUTPUT PARAMETERS:

```
    INTEGER, INTENT(OUT) :: YYYYMMDD   ! Date in YYYY/MM/DD format
    INTEGER, INTENT(OUT) :: HHMMSS     ! Time in hh:mm:ss format
```

REMARKS:

```
   Algorithm taken from "Practical Astronomy With Your Calculator",
   Third Edition, by Peter Duffett-Smith, Cambridge UP, 1992.
```

REVISION HISTORY:

```
  (1 ) Now compute HHMMSS correctly.  Also use REAL*4 variables HH, MM, SS
        to avoid roundoff errors. (bmy, 11/21/01)
  (2 ) Renamed NYMD to YYYYMMDD and NHMS to HHMMSS for documentation
        purposes (bmy, 6/26/02)
  20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.9 Fortran: Module Interface pressure_mod.f

Module PRESSURE_MOD contains variables and routines which specify the grid box pressures for both hybrid or pure-sigma models. This is necessary for running GEOS-Chem with the GEOS-4 or GEOS-5 hybrid grids.

INTERFACE:

```
    MODULE PRESSURE_MOD
```

USES:

```
    IMPLICIT NONE
    PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
    PUBLIC :: CLEANUP_PRESSURE
    PUBLIC :: GET_AP
    PUBLIC :: GET_BP
    PUBLIC :: GET_PCENTER
    PUBLIC :: GET_PEDGE
    PUBLIC :: INIT_PRESSURE
    PUBLIC :: SET_FLOATING_PRESSURE
```

REMARKS:

```
  Hybrid Grid Coordinate Definition: (dsa, bmy, 8/27/02, 8/13/10)
  ============================================================================
                                                                             .
  GEOS-4, GEOS-5, and MERRA (hybrid grids):
  ----------------------------------------------------------------------------
  For GEOS-4/GEOS-5/MERRA met data products, the pressure at the bottom edge
  of grid box (I,J,L) is defined as follows:
```

```
                                                                          .
   Pedge(I,J,L) = Ap(L) + [ Bp(L) * Psurface(I,J) ]
                                                                          .
where
                                                                          .
   Psurface(I,J) is  the "true" surface pressure at lon,lat (I,J)
   Ap(L)         has the same units as surface pressure [hPa]
   Bp(L)         is  a unitless constant given at level edges
                                                                          .
Ap(L) and Bp(L) are given to us by GMAO.
                                                                          .
                                                                          .
GEOS-3 (pure-sigma) and GCAP (hybrid grid):
============================================================================
GEOS-3 is a pure-sigma grid.  GCAP is a hybrid grid, but its grid is
defined as if it were a pure sigma grid (i.e. PTOP=150 hPa, and negative
sigma edges at higher levels).  For these grids, can stil use the same
formula as for GEOS-4/GEOS-5/MERRA, with one modification:
                                                                          .
   Pedge(I,J,L) = Ap(L) + [ Bp(L) * ( Psurface(I,J) - PTOP ) ]
                                                                          .
where
                                                                          .
   Psurface(I,J) = the "true" surface pressure at lon,lat (I,J)
   Ap(L)         = PTOP    = model top pressure
   Bp(L)         = SIGE(L) = bottom sigma edge of level L
                                                                          .
                                                                          .
The following are true for GCAP, GEOS-3, GEOS-4, GEOS-5, MERRA:
============================================================================
(1) Bp(LLPAR+1) = 0.0          (L=LLPAR+1 is the atmosphere top)
(2) Bp(1)       = 1.0          (L=1       is the surface       )
(3) PTOP        = Ap(LLPAR+1)  (L=LLPAR+1 is the atmosphere top)
```

REVISION HISTORY:

```
27 Aug 2002 - D. Abbot & R. Yantosca - Initial version
(1 ) Be sure to check PFLT for NaN or Infinities (bmy, 8/27/02)
(2 ) Updated comments (bmy, 5/8/03)
(3 ) Updated format string for fvDAS (bmy, 6/19/03)
(4 ) Bug fix: use PFLT instead of PFLT-PTOP for GEOS-4 (bmy, 10/24/03)
(5 ) Modifications for 30L and 55L GEOS-4 grids (bmy, 11/3/03)
(6 ) Added parallel DO-loop in SET_FLOATING_PRESSURE (bmy, 4/14/04)
(7 ) Modified for GCAP and GEOS-5 grids (swu, bmy, 5/24/05)
(8 ) Removed obsolete reference to "CMN" (bmy, 4/25/06)
(9 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
(10) Added Ap and Bp for GEOS-5 met fields (bmy, 10/30/07)
20 Nov 2009 - R. Yantosca - Added ProTeX headers
13 Aug 2010 - R. Yantosca - Added modifications for MERRA met fields
```

```
    30 Aug 2010 - R. Yantosca - Updated comments
```

1.9.1 get_ap

Function GET_AP returns the ”A” term [hPa] for the hybrid ETA coordinate.

INTERFACE:

```
      FUNCTION GET_AP( L ) RESULT( AP_TEMP )
```

USES:

```
#     include "CMN_SIZE"             ! Size parameters
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: L       ! GEOS-Chem level index
```

RETURN VALUE:

```
      REAL*8              :: AP_TEMP  ! Corresponding "A" value [hPa]
                                      !  at bottom edge of level L
```

REVISION HISTORY:

```
    20 Aug 2002 - D. Abbot & R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.9.2 get_bp

Function GET_BP returns the ”B” term [unitless] for the hybrid ETA coordinate.

INTERFACE:

```
      FUNCTION GET_BP( L ) RESULT( BP_TEMP )
```

USES:

```
#     include "CMN_SIZE"             ! Size parameters
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: L       ! GEOS-Chem level index
```

RETURN VALUE:

```
      REAL*8              :: BP_TEMP  ! Corresponding "B" value [unitless]
                                      !  at bottom edge of level L
```

REVISION HISTORY:

```
    20 Aug 2002 - D. Abbot & R. Yantosca - Initial version
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.9.3 set_floating_pressure

Subroutine SET_FLOATING_PRESSURE initializes the floating pressure field PFLT with a pressure from the main program. This is needed to initialize and reset PFLT after transport.

INTERFACE:

```
      SUBROUTINE SET_FLOATING_PRESSURE( PS )
```

USES:

```
      USE ERROR_MOD, ONLY : CHECK_VALUE

#     include "CMN_SIZE"  ! Size parameters
```

INPUT PARAMETERS:

```
      ! Array containing pressure with which to initialize PFLT [hPa]
      REAL*8, INTENT(IN) :: PS(IIPAR,JJPAR)
```

REVISION HISTORY:

```
   27 Aug 2002 - D. Abbot, B. Field,  R. Yantosca - Initial version
   (1 ) Now check PFLT for NaN or Infinities (bmy, 8/27/02)
   (2 ) Added parallel DO-loop (bmy, 4/14/04)
   20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.9.4 get_pedge

Function GET_PEDGE returns the pressure at the bottom edge of level L. L=1 is the surface, L=LLPAR+1 is the atm top.

INTERFACE:

```
      FUNCTION GET_PEDGE( I, J, L ) RESULT( PEDGE )
```

USES:

```
#     include "CMN_SIZE"   ! PTOP
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: I      ! GEOS-Chem lon   index
      INTEGER, INTENT(IN) :: J      ! GEOS-Chem lat   index
      INTEGER, INTENT(IN) :: L      ! GEOS-Chem level index
```

RETURN VALUE:

```
      REAL*8             :: PEDGE  ! Pressure @ bottom edge of (I,J,L) [hPa]
```

REVISION HISTORY:

```
    20 Aug 2002 - D. Abbot & R. Yantosca - Initial version
    (1 ) Bug fix: use PFLT instead of PFLT-PTOP for GEOS-4 (bmy, 10/24/03)
    (2 ) Now treat GEOS-5 the same way as GEOS-4 (bmy, 10/30/07)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
    13 Aug 2010 - R. Yantosca - Compute PEDGE for MERRA the same as for GEOS-5
```

1.9.5 get_pcenter

Function GET_PCENTER returns the pressure at the vertical midpoint of level L.

INTERFACE:

```
      FUNCTION GET_PCENTER( I, J, L ) RESULT( PCENTER )
```

USES:

```
#     include "CMN_SIZE"   ! PTOP
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: I        ! GEOS-Chem lon   index
      INTEGER, INTENT(IN) :: J        ! GEOS-Chem lat   index
      INTEGER, INTENT(IN) :: L        ! GEOS-Chem level index
```

RETURN VALUE:

```
      REAL*8              :: PCENTER  ! Pressure @ center of (I,J,L) [hPa]
```

REVISION HISTORY:

```
    20 Aug 2002 - D. Abbot & R. Yantosca - Initial version
    (1 ) Updated format string for fvDAS (bmy, 6/19/03)
    (2 ) Removed reference to "CMN", it's obsolete (bmy, 4/25/06)
    20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.9.6 init_pressure

Subroutine INIT_PRESSURE allocates and initializes the AP and BP arrays. It must be called in "main.f", after SIGE is defined. GEOS-4 and GEOS-5 requires the hybrid pressure system specified by the listed values of AP and BP, while earlier versions of GEOS use a pure sigma pressure system. GCAP met fields (based on GISS) also use a hybrid system.

INTERFACE:

```
      SUBROUTINE INIT_PRESSURE
```

USES:

```
      ! References to F90 modules
      USE ERROR_MOD, ONLY : ALLOC_ERR

#     include "CMN_SIZE"  ! LLPAR, PTOP
```

REVISION HISTORY:

```
27 Aug 2002 - D. Abbot, S. Wu, & R. Yantosca - Initial version
(1 ) Now reference ALLOC_ERR from "error_mod.f" (bmy, 10/15/02)
(2 ) Now echo Ap, Bp to std output (bmy, 3/14/03)
(3 ) Now print LLPAR+1 levels for Ap, Bp.  Remove reference to SIGE, it's
      obsolete.  Also now use C-preprocessor switch GRID30LEV instead of
      IF statements to define vertical coordinates. (bmy, 11/3/03)
(4 ) Now modified for both GCAP & GEOS-5 vertical grids (swu, bmy, 5/24/05)
(5 ) Renamed GRID30LEV to GRIDREDUCED (bmy, 10/30/07)
20 Nov 2009 - R. Yantosca - Added ProTeX header
13 Aug 2010 - R. Yantosca - Compute Ap and Bp for MERRA the same way as for
                            GEOS-5.  The vertical grids are identical.
30 Aug 2010 - R. Yantosca - Updated comments
30 Nov 2010 - R. Yantosca - Further improved comments about how GEOS-4 and
                            GEOS-5 vertical levels are lumped together.
```

1.9.7 cleanup_pressure

Subroutine CLEANUP_PRESSURE deallocates all allocated arrays at the end of a GEOS-Chem model run.

INTERFACE:

```
SUBROUTINE CLEANUP_PRESSURE
```

REVISION HISTORY:

```
20 Aug 2002 - D. Abbot & R. Yantosca - Initial version
20 Nov 2009 - R. Yantosca - Added ProTeX header
```

1.10 Fortran: Module Interface time_mod

Module TIME_MOD contains GEOS-Chem date and time variables and timesteps, and routines for accessing them.

INTERFACE:

```
MODULE TIME_MOD
```

USES:

```
IMPLICIT NONE
PRIVATE
```

PUBLIC MEMBER FUNCTIONS:

```
      PUBLIC  :: SET_CURRENT_TIME
      PUBLIC  :: SET_BEGIN_TIME
      PUBLIC  :: SET_END_TIME
      PUBLIC  :: SET_NDIAGTIME
      PUBLIC  :: SET_DIAGb
      PUBLIC  :: SET_DIAGe
      PUBLIC  :: SET_TIMESTEPS
      PUBLIC  :: SET_CT_CHEM
      PUBLIC  :: SET_CT_CONV
      PUBLIC  :: SET_CT_DYN
      PUBLIC  :: SET_CT_EMIS
      PUBLIC  :: SET_CT_DIAG
      PUBLIC  :: SET_CT_A1
      PUBLIC  :: SET_CT_A3
      PUBLIC  :: SET_CT_A6
      PUBLIC  :: SET_CT_I6
      PUBLIC  :: SET_CT_XTRA
      PUBLIC  :: SET_ELAPSED_MIN
      PUBLIC  :: GET_JD
      PUBLIC  :: GET_ELAPSED_MIN
      PUBLIC  :: GET_ELAPSED_SEC
      PUBLIC  :: GET_NYMDb
      PUBLIC  :: GET_NHMSb
      PUBLIC  :: GET_NYMDe
      PUBLIC  :: GET_NHMSe
      PUBLIC  :: GET_NYMD
      PUBLIC  :: GET_NHMS
      PUBLIC  :: GET_NDIAGTIME
      PUBLIC  :: GET_TIME_AHEAD
      PUBLIC  :: GET_MONTH
      PUBLIC  :: GET_DAY
      PUBLIC  :: GET_YEAR
      PUBLIC  :: GET_HOUR
      PUBLIC  :: GET_MINUTE
      PUBLIC  :: GET_SECOND
      PUBLIC  :: GET_DAY_OF_YEAR
      PUBLIC  :: GET_DAY_OF_WEEK
      PUBLIC  :: GET_GMT
      PUBLIC  :: GET_TAU
      PUBLIC  :: GET_TAUb
      PUBLIC  :: GET_TAUe
      PUBLIC  :: GET_DIAGb
      PUBLIC  :: GET_DIAGe
      PUBLIC  :: GET_LOCALTIME
      PUBLIC  :: GET_SEASON
      PUBLIC  :: GET_TS_CHEM
      PUBLIC  :: GET_TS_CONV
      PUBLIC  :: GET_TS_DIAG
      PUBLIC  :: GET_TS_DYN
      PUBLIC  :: GET_TS_EMIS
      PUBLIC  :: GET_TS_UNIT
      PUBLIC  :: GET_TS_SUN_2
      PUBLIC  :: GET_CT_CHEM
      PUBLIC  :: GET_CT_CONV
      PUBLIC  :: GET_CT_DYN
      PUBLIC  :: GET_CT_EMIS
      PUBLIC  :: GET_CT_A1
      PUBLIC  :: GET_CT_A3
      PUBLIC  :: GET_CT_A6
      PUBLIC  :: GET_CT_I6
      PUBLIC  :: GET_CT_XTRA
      PUBLIC  :: GET_CT_DIAG
      PUBLIC  :: GET_A1_TIME
      PUBLIC  :: GET_A3_TIME
      PUBLIC  :: GET_A6_TIME
      PUBLIC  :: GET_I6_TIME
      PUBLIC  :: GET_FIRST_A1_TIME
      PUBLIC  :: GET_FIRST_A3_TIME
      PUBLIC  :: GET_FIRST_A6_TIME
      PUBLIC  :: GET_FIRST_I6_TIME
      PUBLIC  :: ITS_TIME_FOR_CHEM
      PUBLIC  :: ITS_TIME_FOR_CONV
      PUBLIC  :: ITS_TIME_FOR_DYN
      PUBLIC  :: ITS_TIME_FOR_EMIS
      PUBLIC  :: ITS_TIME_FOR_UNIT
      PUBLIC  :: ITS_TIME_FOR_DIAG
      PUBLIC  :: ITS_TIME_FOR_A1
      PUBLIC  :: ITS_TIME_FOR_A3
      PUBLIC  :: ITS_TIME_FOR_A6
      PUBLIC  :: ITS_TIME_FOR_I6
      PUBLIC  :: ITS_TIME_FOR_UNZIP
      PUBLIC  :: ITS_TIME_FOR_DEL
      PUBLIC  :: ITS_TIME_FOR_EXIT
      PUBLIC  :: ITS_TIME_FOR_BPCH
      PUBLIC  :: ITS_A_LEAPYEAR
      PUBLIC  :: ITS_A_NEW_YEAR
      PUBLIC  :: ITS_A_NEW_MONTH
      PUBLIC  :: ITS_MIDMONTH
      PUBLIC  :: ITS_A_NEW_DAY
      PUBLIC  :: ITS_A_NEW_SEASON
      PUBLIC  :: PRINT_CURRENT_TIME
      PUBLIC  :: TIMESTAMP_STRING
      PUBLIC  :: YMD_EXTRACT
      PUBLIC  :: EXPAND_DATE
      PUBLIC  :: SYSTEM_DATE_TIME
      PUBLIC  :: SYSTEM_TIMESTAMP
```

```
    PUBLIC  :: TIMESTAMP_DIAG
    PUBLIC  :: GET_NYMD_DIAG
```

REVISION HISTORY:

```
  21 Jun 2000 - R. Yantosca - Initial version
  (1 ) Updated comments (bmy, 9/4/01)
  (2 ) Added routine YMD_EXTRACT.  Also rewrote TIMECHECK using astronomical
        Julian day routines from "julday_mod.f". (bmy, 11/21/01)
  (3 ) Eliminated obsolete code (bmy, 2/27/02)
  (4 ) Updated comments (bmy, 5/28/02)
  (5 ) Added routine "expand_date".  Also now reference "charpak_mod.f".
        (bmy, 6/27/02)
  (6 ) Now references "error_mod.f".  Also added function GET_SEASON, which
        returns the current season number. (bmy, 10/22/02)
  (7 ) Now added module variables and various GET_ and SET_ routines to
        access them.  Now minutes are the smallest timing unit. (bmy, 3/21/03)
  (8 ) Bug fix in DATE_STRING (bmy, 5/15/03)
  (9 ) Added GET_FIRST_A3_TIME and GET_FIRST_A6_TIME.  Also added changes for
        reading fvDAS fields. (bmy, 6/26/03)
  (10) Now allow ITS_A_LEAPYEAR to take an optional argument.  Bug fix for
        Linux: must use ENCODE to convert numbers to strings (bmy, 9/29/03)
  (11) Bug fix in EXPAND_DATE.  Also add optional arguments to function
        TIMESTAMP_STRNIG. (bmy, 10/28/03)
  (12) Changed the name of some cpp switches in "define.h" (bmy, 12/2/03)
  (13) Modified ITS_TIME_FOR_A6 and GET_FIRST_A6_TIME for both GEOS-4
        "a_llk_03" and "a_llk_04" data versions. (bmy, 3/22/04)
  (14) Added routines ITS_A_NEW_MONTH, ITS_A_NEW_YEAR, ITS_A_NEW_DAY.
        (bmy, 4/1/04)
  (15) Added routines ITS_A_NEW_SEASON, GET_NDIAGTIME, SET_NDIAGTIME, and
        variable NDIAGTIME. (bmy, 7/20/04)
  (17) Added routine GET_DAY_OF_WEEK (bmy, 11/5/04)
  (18) Removed obsolete FIRST variable in GET_A3_TIME (bmy, 12/10/04)
  (19) Added routines SYSTEM_DATE_TIME and SYSTEM_TIMESTAMP.  Also modified
        for GCAP and GEOS-5 met fields. (swu, bmy, 5/3/05)
  (20) GCAP/GISS met fields don't have leap years (swu, bmy, 8/29/05)
  (21) Added counter variable & routines for XTRA fields (tmf, bmy, 10/20/05)
  (22) Bug fix in ITS_A_NEW_YEAR (bmy, 11/1/05)
  (23) Added function ITS_MIDMONTH.  Also removed obsolete functions
        NYMD_Y2K, NYMD6_2_NYMD8, NYMD_STRING, DATE_STRING.
        (sas, cdh, bmy, 12/15/05)
  (24) GCAP bug fix: There are no leapyears, so transition from 2/28 to 3/1,
        skipping 2/29 for all years. (swu, bmy, 4/24/06)
  (25) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
  (26) Further bug fix to skip over Feb 29th in GCAP (phs, bmy, 10/3/06)
  (27) Moved ITS_TIME_FOR_BPCH here from "main.f" (bmy, 2/2/07)
  (28) Add TS_DIAG and CT_DIAG variables to correctly output diagnostics
        (good time step).
       Add SET_CT_DIAG and GET_CT_DIAG to implement TS_DIAG correctly.
```

```
         (ccc, 5/21/09)
    (29) Add NYMD_DIAG, GET_NYMD_DIAG, TIMESTAMP_DIAG to get the good timestamp
         for diagnostic filenames (ccc, 8/12/09)
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
    27 Apr 2010 - R. Yantosca - Added OFFSET argument to GET_LOCALTIME
    27 Apr 2010 - R. Yantosca - Added TS_SUN_2 to hold 1/2 of the interval
                                for computing SUNCOS.
    27 Apr 2010 - R. Yantosca - Added public routine GET_TS_SUN_2
    19 Aug 2010 - R. Yantosca - Added variable CT_A1 and routine SET_CT_A1
    20 Aug 2010 - R. Yantosca - Added function ITS_TIME_FOR_A1
    27 Sep 2010 - R. Yantosca - Added function GET_FIRST_I6_TIME
    17 Dec 2010 - R. Yantosca - Bug fix for HHMMSS=240000 in GET_TIME_AHEAD
```

1.10.1 set_current_time

Subroutine SET_CURRENT_TIME takes in the elapsed time in minutes since the start of a GEOS-Chem simulation and sets the GEOS-Chem time variables accordingly.

INTERFACE:

```
      SUBROUTINE SET_CURRENT_TIME
```

USES:

```
      USE JULDAY_MOD, ONLY : JULDAY, CALDATE

#     include "define.h"
```

REVISION HISTORY:

```
    05 Feb 2006 - R. Yantosca - Initial Version
    (1 ) GCAP/GISS fields don't have leap years, so if JULDAY says it's
          Feb 29th, reset MONTH, DAY, JD1 to Mar 1st. (swu, bmy, 8/29/05)
    (2 ) Now references "define.h".  Now add special handling to skip from
          Feb 28th to Mar 1st for GCAP model. (swu, bmy, 4/24/06)
    (3 ) Fix bug in case of GCAP fields for runs that start during leap year
         and after February 29 (phs, 9/27/06)
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.2 set_begin_time

Subroutine SET_BEGIN_TIME initializes NYMDb, NHMSb, and TAUb, which are the YYYYMMDD, HHMMSS, and hours since 1/1/1985 corresponding to the beginning date and time of a GEOS-Chem run.

INTERFACE:

```
      SUBROUTINE SET_BEGIN_TIME( THISNYMDb, THISNHMSb )
```

USES:

```
      USE ERROR_MOD,  ONLY : ERROR_STOP
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: THISNYMDb   ! YYYYMMDD @ start of G-C simulation
      INTEGER, INTENT(IN) :: THISNHMSb   ! HHMMSS   @ start of G-C simulation
```

REVISION HISTORY:

```
   20 Jul 2004 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   16 Dec 2010 - R. Yantosca - Updated error check for THISNYMDe, since
                              MERRA met data goes back prior to 1985
```

1.10.3 set_end_time

Subroutine SET_END_TIME initializes NYMDe, NHMSe, and TAUe, which are the YYYYMMDD, HHMMSS, and hours since 1/1/1985 corresponding to the ending date and time of a GEOS-Chem run.

INTERFACE:

```
      SUBROUTINE SET_END_TIME( THISNYMDe, THISNHMSe )
```

USES:

```
      USE ERROR_MOD,  ONLY : ERROR_STOP
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: THISNYMDe   ! YYYYMMDD @ end of G-C simulation
      INTEGER, INTENT(IN) :: THISNHMSe   ! HHMMSS   @ end of G-C simulation
```

REVISION HISTORY:

```
   20 Jul 2004 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   16 Dec 2010 - R. Yantosca - Updated error check for THISNYMDe, since
                              MERRA met data goes back prior to 1985
```

1.10.4 set_ndiagtime

SET_NDIAGTIME initializes NDIAGTIME, the time of day at which the binary punch file will be written out to disk.

INTERFACE:

```
      SUBROUTINE SET_NDIAGTIME( THIS_NDIAGTIME )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: THIS_NDIAGTIME  ! Initial NDIAGTIMEe [hrs]
```

REVISION HISTORY:

```
   20 Jul 2004 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.5 set_diagb

Subroutine SET_DIAGb initializes DIAGb, the TAU value at the start of the diagnostic averaging interval.

INTERFACE:

```
      SUBROUTINE SET_DIAGb( THISDIAGb )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: THISDIAGb  ! Initial DIAGb value [hrs from 1/1/85]
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.6 set_diage

Subroutine SET_DIAGe initializes DIAGe, the TAU value at the end of the diagnostic averaging interval.

INTERFACE:

```
      SUBROUTINE SET_DIAGe( THISDIAGe )
```

INPUT PARAMETERS:

```
      REAL*8, INTENT(IN) :: THISDIAGe  ! Initial DIAGe value [hrs from 1/1/85]
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.7 set_timesteps

Subroutine SET_TIMESTEPS initializes the timesteps for dynamics, convection, chemistry, emissions, and diagnostics. Counters are also zeroed.

INTERFACE:

```
      SUBROUTINE SET_TIMESTEPS( CHEMISTRY, CONVECTION, DYNAMICS,
     &                          EMISSION,  UNIT_CONV,  DIAGNOS,
     &                          SUNCOS )
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: CHEMISTRY    ! Chemistry  timestep [min]
      INTEGER, INTENT(IN) :: CONVECTION   ! Convection timestep [min]
      INTEGER, INTENT(IN) :: DYNAMICS     ! Dynamic    timestep [min]
      INTEGER, INTENT(IN) :: EMISSION     ! Emission   timestep [min]
      INTEGER, INTENT(IN) :: UNIT_CONV    ! Unit conve timestep [min]
      INTEGER, INTENT(IN) :: DIAGNOS      ! Diagnostic timestep [min]
      INTEGER, INTENT(IN) :: SUNCOS       ! 1/2 of timestep for SUNCOS [min]
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   (1 ) Suppress some output lines (bmy, 7/20/04)
   (2 ) Also zero CT_XTRA (tmf, bmy, 10/20/05)
   (3 ) Add TS_DIAG as the diagnostic timestep. (ccc, 5/13/09)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   27 Apr 2010 - R. Yantosca - Now add SUNCOS argument to set 1/2 of the
                               interval for computing the cosine of the
                               solar zenith angle.
```

1.10.8 set_ct_chem

Subroutine SET_CT_CHEM increments CT_CHEM, the counter of chemistry timesteps executed thus far.

INTERFACE:

```
      SUBROUTINE SET_CT_CHEM( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
      LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
      LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
   21 Mar 2009 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.9 set_ct_conv

Subroutine SET_CT_CONV increments CT_CONV, the counter of convection timesteps executed thus far.

INTERFACE:

```
SUBROUTINE SET_CT_CONV( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
21 Mar 2009 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.10 set_ct_dyn

Subroutine SET_CT_DYN increments CT_DYN, the counter of dynamical timesteps executed thus far.

INTERFACE:

```
SUBROUTINE SET_CT_DYN( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
21 Mar 2009 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.11 set_ct_emis

Subroutine SET_CT_EMIS increments CT_EMIS, the counter of emission timesteps executed thus far.

INTERFACE:

```
SUBROUTINE SET_CT_EMIS( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
       LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
       LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
    21 Mar 2009 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.12 set_ct_diag

Subroutine SET_CT_DIAG increments CT_DIAG, the counter of largest timesteps executed thus far.

INTERFACE:

```
       SUBROUTINE SET_CT_DIAG( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
       LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
       LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
    13 May 2009 - C. Carouge  - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.13 set_ct_a1

Subroutine SET_CT_A1 increments CT_A1, the counter of the number of times we have read in A1 fields.

INTERFACE:

```
       SUBROUTINE SET_CT_A1( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
       LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
       LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
    19 Aug 2010 - R. Yantosca - Initial version
```

1.10.14 set_ct_a3

Subroutine SET_CT_A3 increments CT_A3, the counter of the number of times we have read in A-3 fields.

INTERFACE:

```
      SUBROUTINE SET_CT_A3( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
      LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
      LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.15 set_ct_a6

Subroutine SET_CT_A6 increments CT_A6, the counter of the number of times we have read in A-6 fields.

INTERFACE:

```
      SUBROUTINE SET_CT_A6( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
      LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
      LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.16 set_ct_i6

Subroutine SET_CT_I6 increments CT_I6, the counter of the number of times we have read in I-6 fields.

INTERFACE:

```
      SUBROUTINE SET_CT_I6( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
      LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
      LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.17 set_ct_xtra

Subroutine SET_CT_XTRA increments CT_XTRA, the counter of the number of times we have read in GEOS-3 XTRA fields.

INTERFACE:

```
      SUBROUTINE SET_CT_XTRA( INCREMENT, RESET )
```

INPUT PARAMETERS:

```
      LOGICAL, INTENT(IN), OPTIONAL :: INCREMENT  ! Increment counter?
      LOGICAL, INTENT(IN), OPTIONAL :: RESET      ! Reset counter?
```

REVISION HISTORY:

```
    20 Oct 2009 - T-M Fu, R. Yantosca - Initial Version
    15 Jan 2010 -         R. Yantosca - Added ProTeX headers
```

1.10.18 set_elapsed_min

Subroutine SET_ELAPSED_MIN increments the number of elapsed minutes by the dynamic timestep TS_DYN.

INTERFACE:

```
      SUBROUTINE SET_ELAPSED_MIN
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.19 get_jd

Function GET_JD is a wrapper for the JULDAY routine. Given the current NYMD and NHMS values, GET_JD will return the current astronomical Julian date.

INTERFACE:

```
FUNCTION GET_JD( THISNYMD, THISNHMS ) RESULT( THISJD )
```

USES:

```
USE JULDAY_MOD, ONLY : JULDAY
```

INPUT PARAMETERS:

```
INTEGER, INTENT(IN)  :: THISNYMD   ! YYYY/MM/DD value
INTEGER, INTENT(IN)  :: THISNHMS   ! hh:mm:ss   value
```

RETURN VALUE:

```
REAL*8               :: THISJD     ! Output value
```

REVISION HISTORY:

```
05 Feb 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.20 get_elapsed_min

Function GET_ELAPSED_MIN returns the elapsed minutes since the start of a GEOS-chem run.

INTERFACE:

```
FUNCTION GET_ELAPSED_MIN() RESULT( THIS_ELAPSED_MIN )
```

RETURN VALUE:

```
INTEGER :: THIS_ELAPSED_MIN
```

REVISION HISTORY:

```
05 Feb 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.21 get_elapsed_sec

Function GET_ELAPSED_SEC returns the elapsed minutes since the start of a GEOS-Chem run to the calling program.

INTERFACE:

```
      FUNCTION GET_ELAPSED_SEC() RESULT( THIS_ELAPSED_SEC )
```

RETURN VALUE:

```
      INTEGER :: THIS_ELAPSED_SEC
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.22 get_nymdb

Function GET_NYMDb returns the NYMDb value (YYYYMMDD at the beginning of the run).

INTERFACE:

```
      FUNCTION GET_NYMDb() RESULT( THISNYMDb )
```

RETURN VALUE:

```
      INTEGER :: THISNYMDb
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.23 get_nhmsb

Function GET_NHMSb returns the NHMSb value (HHMMSS at the beginning of the run) to the calling program. (bmy, 3/21/03)

INTERFACE:

```
      FUNCTION GET_NHMSb() RESULT( THISNHMSb )
```

RETURN VALUE:

```
      INTEGER :: THISNHMSb
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.24 get_nymde

Function GET_NYMDe returns the NYMDe value (YYYYMMDD at the end of the run) to the calling program. (bmy, 3/21/03)

INTERFACE:

```
      FUNCTION GET_NYMDe() RESULT( THISNYMDe )
```

RETURN VALUE:

```
      INTEGER :: THISNYMDe
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.25 get_nhmse

Function GET_NHMSe returns the NHMSe value (HHMMSS at the end of the run).

INTERFACE:

```
      FUNCTION GET_NHMSe() RESULT( THISNHMSe )
```

RETURN VALUE:

```
      INTEGER :: THISNHMSe
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.26 get_nymd

Function GET_NYMD returns the current NYMD value (YYYYMMDD).

INTERFACE:

```
      FUNCTION GET_NYMD() RESULT( THISNYMD )
```

RETURN VALUE:

```
      INTEGER :: THISNYMD
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.27 get_nhms

Function GET_NHMS returns the current NHMS value (HHMMSS).

INTERFACE:

```
      FUNCTION GET_NHMS() RESULT( THISNHMS )
```

RETURN VALUE:

```
      INTEGER :: THISNHMS
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.28 get_ndiagtime

Subroutine GET_NDIAGTIME returns to the calling program NDIAGTIME, the time of day at which the binary punch file will be written out to disk.

INTERFACE:

```
      FUNCTION GET_NDIAGTIME() RESULT( THIS_NDIAGTIME )
```

RETURN VALUE:

```
      INTEGER :: THIS_NDIAGTIME
```

REVISION HISTORY:

```
   20 Jul 2004 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.29 get_time_ahead

Function GET_3h_AHEAD returns to the calling program a 2-element vector containing the YYYYMMDD and HHMMSS values at the current time plus N_MINS minutes.

INTERFACE:

```
      FUNCTION GET_TIME_AHEAD( N_MINS ) RESULT( DATE )
```

USES:

```
      USE JULDAY_MOD, ONLY : CALDATE

#     include "define.h"   ! C-preprocessor flags
```

INPUT PARAMETERS:

```
INTEGER, INTENT(IN) :: N_MINS   ! Minutes ahead to compute date & time
```

RETURN VALUE:

```
INTEGER             :: DATE(2)  ! Date & time output
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
(1 ) Bug fix for GCAP leap year case (phs, bmy, 12/8/06)
15 Jan 2010 - R. Yantosca - Added ProTeX headers
17 Dec 2010 - R. Yantosca - Added fix in case HHMMSS is returned as 240000
```

1.10.30 get_month

Function GET_MONTH returns the current GMT month.

INTERFACE:

```
FUNCTION GET_MONTH() RESULT( THISMONTH )
```

RETURN VALUE:

```
INTEGER :: THISMONTH
```

REVISION HISTORY:

```
05 Feb 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.31 get_day

Function GET_DAY returns the current GMT day.

INTERFACE:

```
FUNCTION GET_DAY() RESULT( THISDAY )
```

RETURN VALUE:

```
INTEGER :: THISDAY
```

REVISION HISTORY:

```
05 Feb 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.32 get_year

Function GET_YEAR returns the current GMT year.

INTERFACE:

```
      FUNCTION GET_YEAR() RESULT( THISYEAR )
```

RETURN VALUE:

```
      INTEGER :: THISYEAR
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.33 get_hour

Function GET_HOUR returns the current GMT hour.

INTERFACE:

```
      FUNCTION GET_HOUR() RESULT( THISHOUR )
```

RETURN VALUE:

```
      INTEGER :: THISHOUR
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.34 get_minute

Function GET_MINUTE returns the current GMT minutes.

INTERFACE:

```
      FUNCTION GET_MINUTE() RESULT( THISMINUTE )
```

RETURN VALUE:

```
      INTEGER :: THISMINUTE
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.35 get_second

Function GET_SECOND returns the current GMT seconds. calling program.

INTERFACE:

```
      FUNCTION GET_SECOND() RESULT( THISSECOND )
```

RETURN VALUE:

```
      INTEGER :: THISSECOND
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.36 get_day_of_year

Function GET_DAY_OF_YEAR returns the current day of the year (0-365 or 0-366 for leap years) to the calling program.

INTERFACE:

```
      FUNCTION GET_DAY_OF_YEAR() RESULT( THISDAYOFYEAR )
```

RETURN VALUE:

```
      INTEGER :: THISDAYOFYEAR  ! Day of year
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.37 get_day_of_week

Function GET_DAY_OF_WEEK returns the day of the week as a number: Sun=0, Mon=1, Tue=2, Wed=3, Thu=4, Fri=5, Sat=6.

INTERFACE:

```
      FUNCTION GET_DAY_OF_WEEK() RESULT( DAY_NUM )
```

USES:

```
      USE JULDAY_MOD, ONLY : JULDAY
```

RETURN VALUE:

```
       INTEGER :: DAY_NUM   ! Day number of week
```

REMARKS:

```
    Reference:
    ----------
    "Practical Astronomy with Your Calculator", 3rd Ed.  Peter Duffett-Smith,
      Cambridge UP, 1992, p9.
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.38 get_gmt

Function GET_GMT returns the current Greenwich Mean Time to the calling program.

INTERFACE:

```
       FUNCTION GET_GMT() RESULT( THISGMT )
```

RETURN VALUE:

```
       REAL*8 :: THISGMT   ! Greenwich mean time [hrs]
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.39 get_tau

Function GET_TAU returns TAU (hours since 1 Jan 1985 at the start of a GEOS-Chem run) to the calling program.

INTERFACE:

```
       FUNCTION GET_TAU() RESULT( THISTAU )
```

RETURN VALUE:

```
       REAL*8 :: THISTAU  ! TAUb [hrs since 1/1/1985]
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.40 get_taub

Function GET_TAUb returns TAUb (hours since 1 Jan 1985 at the start of a GEOS-Chem run) to the calling program.

INTERFACE:

```
      FUNCTION GET_TAUb() RESULT( THISTAUb )
```

RETURN VALUE:

```
      REAL*8 :: THISTAUb  ! TAUb [hrs since 1/1/1985]
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.41 get_taue

Function GET_TAUe returns TAUe (hours since 1 Jan 1985 at the end of a GEOS-Chem run) to the calling program.

INTERFACE:

```
      FUNCTION GET_TAUe() RESULT( THISTAUe )
```

RETURN VALUE:

```
      REAL*8 :: THISTAUe  ! TAUe [hrs since 1/1/1985]
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.42 get_diagb

Function GET_DIAGb returns DIAGb (hours since 1 Jan 1985 at the start of a diagnostic interval) to the calling program.

INTERFACE:

```
      FUNCTION GET_DIAGb() RESULT( THISDIAGb )
```

RETURN VALUE:

```
      INTEGER :: THISDIAGb   ! DIAGb [hrs sincd 1/1/1985]
```

REVISION HISTORY:

```
   05 Feb 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.43 get_diage

Function GET_DIAGe returns DIAGe (hours since 1 Jan 1985 at the end of a diagnostic interval) to the calling program.

INTERFACE:

```
      FUNCTION GET_DIAGe() RESULT( THISDIAGe )
```

RETURN VALUE:

```
      INTEGER :: THISDIAGe   ! DIAGe [hrs sincd 1/1/1985]
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.44 get_localtime

Function GET_LOCALTIME returns the local time of a grid box to the calling program. (bmy, 2/5/03)

INTERFACE:

```
      FUNCTION GET_LOCALTIME( I, OFFSET ) RESULT( THISLOCALTIME )
```

USES:

```
      USE GRID_MOD, ONLY : GET_XMID
```

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN)           :: I        ! Longitude index
      REAL*8,  INTENT(IN), OPTIONAL :: OFFSET   ! Offset to apply to GMT [hrs]
```

RETURN VALUE:

```
      REAL*8                        :: THISLOCALTIME  ! Local time [hrs]
```

REMARKS:

```
    Local Time = GMT + ( longitude / 15 ) since each hour of time
    corresponds to 15 degrees of longitude on the globe
```

REVISION HISTORY:

```
    05 Feb 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
    27 Apr 2010 - R. Yantosca - Add OFFSET to argument list, to allow the
                                local time to be computed at an arbitrary time
                                (e.g. at the halfway point of an interval)
```

1.10.45 get_season

Function GET_SEASON returns the climatological season number (1=DJF, 2=MAM, 3=JJA, 4=SON) to the calling program.

INTERFACE:

```
FUNCTION GET_SEASON() RESULT( THISSEASON )
```

RETURN VALUE:

```
INTEGER :: THISSEASON   ! Current season
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.46 get_ts_chem

Function GET_TS_CHEM returns the chemistry timestep in minutes.

INTERFACE:

```
FUNCTION GET_TS_CHEM() RESULT( THIS_TS_CHEM )
```

RETURN VALUE:

```
INTEGER :: THIS_TS_CHEM   ! ! Chemistry timestep [min]
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.47 get_ts_conv

Function GET_TS_CONV returns the convection timestep in minutes.

INTERFACE:

```
FUNCTION GET_TS_CONV() RESULT( THIS_TS_CONV )
```

RETURN VALUE:

```
INTEGER :: THIS_TS_CONV   ! Convective timestep [min]
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.48 get_ts_diag

Function GET_TS_DIAG returns the diagnostic timestep in minutes.

INTERFACE:

```
      FUNCTION GET_TS_DIAG() RESULT( THIS_TS_DIAG )
```

RETURN VALUE:

```
      INTEGER :: THIS_TS_DIAG   ! Diagnostic timestep [min]
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.49 get_ts_dyn

Function GET_TS_DIAG returns the diagnostic timestep in minutes.

INTERFACE:

```
      FUNCTION GET_TS_DYN() RESULT( THIS_TS_DYN )
```

RETURN VALUE:

```
      INTEGER :: THIS_TS_DYN    ! Dynamic timestep [min]
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.50 get_ts_emis

Function GET_TS_EMIS returns the emission timestep in minutes.

INTERFACE:

```
      FUNCTION GET_TS_EMIS() RESULT( THIS_TS_EMIS )
```

RETURN VALUE:

```
      INTEGER :: THIS_TS_EMIS   ! Emissions timestep [min]
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.51 get_ts_unit

Function GET_TS_UNIT returns the unit-conversion timestep in minutes.

INTERFACE:

```
FUNCTION GET_TS_UNIT() RESULT( THIS_TS_UNIT )
```

RETURN VALUE:

```
INTEGER :: THIS_TS_UNIT   ! Unit conversion timestep [min]
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.52 get_ts_sun_2

Function GET_TS_SUN_2 returns TS_SUN_2, which is 1/2 of the interval at which we are computing the cosine of the solar zenith angle, aka SUNCOS. This is required to move the time at which we compute SUNCOS to the middle of the chemistry timestep interval.

INTERFACE:

```
FUNCTION GET_TS_SUN_2() RESULT( THIS_TS_SUN_2 )
```

RETURN VALUE:

```
INTEGER :: THIS_TS_SUN_2  ! 1/2 of SUNCOS interval [min]
```

REVISION HISTORY:

```
27 Apr 2010 - R. Yantosca - Initial version
```

1.10.53 get_ct_chem

Function GET_CT_CHEM returns the chemistry timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_CHEM() RESULT( THIS_CT_CHEM )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_CHEM
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.54 get_ct_conv

Function GET_CT_CONV returns the convection timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_CONV() RESULT( THIS_CT_CONV )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_CONV   ! # of convection timesteps
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.55 get_ct_dyn

Function GET_CT_CHEM returns the dynamic timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_DYN() RESULT( THIS_CT_DYN )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_DYN   ! # of dynamics timesteps
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.56 get_ct_emis

Function GET_CT_CHEM returns the emissions timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_EMIS() RESULT( THIS_CT_EMIS )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_EMIS  ! # of emissions timesteps
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.57 get_ct_a1

Function GET_CT_A1 returns the A1 fields timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_A1() RESULT( THIS_CT_A1 )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_A1   ! # of A-3 timesteps
```

REVISION HISTORY:

```
19 Aug 2010 - R. Yantosca - Initial version
```

1.10.58 get_ct_a3

Function GET_CT_A3 returns the A-3 fields timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_A3() RESULT( THIS_CT_A3 )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_A3   ! # of A-3 timesteps
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.59 get_ct_a6

Function GET_CT_A6 returns the A-6 fields timestep counter to the calling program.

INTERFACE:

```
FUNCTION GET_CT_A6() RESULT( THIS_CT_A6 )
```

RETURN VALUE:

```
INTEGER :: THIS_CT_A6   ! # of A-6 timesteps
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.60 get_ct_i6

Function GET_CT_I6 returns the I-6 fields timestep counter to the calling program

INTERFACE:

```
      FUNCTION GET_CT_I6() RESULT( THIS_CT_I6 )
```

RETURN VALUE:

```
      INTEGER :: THIS_CT_I6   ! # of I-6 timesteps
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.61 get_ct_xtra

Function GET_CT_XTRA returns the XTRA fields timestep counter to the calling program.

INTERFACE:

```
      FUNCTION GET_CT_XTRA() RESULT( THIS_CT_XTRA )
```

RETURN VALUE:

```
      INTEGER :: THIS_CT_XTRA    ! # of XTRA timesteps
```

REVISION HISTORY:

```
    20 Oct 2005 - T-M Fu, R. Yantosca - Initial Version
    15 Jan 2010 -         R. Yantosca - Added ProTeX headers
```

1.10.62 get_ct_diag

Function GET_CT_DIAG returns the DIAG timestep counter to the calling program.

INTERFACE:

```
      FUNCTION GET_CT_DIAG() RESULT( THIS_CT_DIAG )
```

RETURN VALUE:

```
      INTEGER :: THIS_CT_DIAG   ! # of diagnostic timesteps
  !
```

REVISION HISTORY:

```
    21 May 2009 - C. Carouge  - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.63 get_a1_time

Function GET_A1_TIME returns the correct YYYYMMDD and HHMMSS values that are needed to read in the next average 1-hour (A-1) fields.

INTERFACE:

```
      FUNCTION GET_A1_TIME() RESULT( DATE )
```

USES:

```
 #    include "define.h"
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS values
```

REVISION HISTORY:

```
    19 Aug 2010 - R. Yantosca - Initial version
```

1.10.64 get_a3_time

Function GET_A3_TIME returns the correct YYYYMMDD and HHMMSS values that are needed to read in the next average 3-hour (A-3) fields.

INTERFACE:

```
      FUNCTION GET_A3_TIME() RESULT( DATE )
```

USES:

```
 #    include "define.h"
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS values
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    (1 ) Now return proper time for GEOS-4/fvDAS fields (bmy, 6/19/03)
    (2 ) Remove reference to FIRST variable (bmy, 12/10/04)
    (3 ) Now modified for GCAP and GEOS-5 met fields (swu, bmy, 5/24/05)
    (4 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)

    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.65 get_a6_time

Function GET_A6_TIME returns the correct YYYYMMDD and HHMMSS values that are needed to read in the next average 6-hour (A-6) fields.

INTERFACE:

```
      FUNCTION GET_A6_TIME() RESULT( DATE )
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS time
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.66 get_i6_time

Function GET_I6_TIME returns the correct YYYYMMDD and HHMMSS values that are needed to read in the next instantaneous 6-hour (I-6) fields.

INTERFACE:

```
      FUNCTION GET_I6_TIME() RESULT( DATE )
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS values
```

REMARKS:

```
   Modified for start times other than 0 GMT.  However someone should check
   to make sure it works properly for the GCAP simulation. (bmy, 9/27/10)
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   (1 ) Bug fix for GCAP: skip over Feb 29th (no leapyears). (bmy, 4/24/06)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   27 Sep 2010 - R. Yantosca - Now works for start times other than 0 GMT
```

1.10.67 get_first_a1_time

Function GET_FIRST_A1_TIME returns the correct YYYYMMDD and HHMMSS values the first time that A-3 fields are read in from disk.

INTERFACE:

```
      FUNCTION GET_FIRST_A1_TIME() RESULT( DATE )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS values
```

REVISION HISTORY:

```
   26 Jun 2003 - R. Yantosca - Initial Version
   (1 ) Now modified for GCAP and GEOS-5 data (swu, bmy, 5/24/05)
   (2 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.68 get_first_a3_time

Function GET_FIRST_A3_TIME returns the correct YYYYMMDD and HHMMSS values the first time that A-3 fields are read in from disk.

INTERFACE:

```
      FUNCTION GET_FIRST_A3_TIME() RESULT( DATE )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
      INTEGER :: DATE(2)   ! YYYYMMDD and HHMMSS values
```

REVISION HISTORY:

```
   26 Jun 2003 - R. Yantosca - Initial Version
   (1 ) Now modified for GCAP and GEOS-5 data (swu, bmy, 5/24/05)
   (2 ) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   27 Sep 2010 - R. Yantosca - Modified for start times other than 0 GMT
```

1.10.69 get_first_a6_time

Function GET_FIRST_A6_TIME returns the correct YYYYMMDD and HHMMSS values the first time that A-6 fields are read in from disk.

INTERFACE:

```
      FUNCTION GET_FIRST_A6_TIME() RESULT( DATE )
```

USES:

```
#     include "define.h"
```

RETURN VALUE:

```
      INTEGER :: DATE(2)    ! YYYYMMDD, HHMMSS values
```

REVISION HISTORY:

```
   26 Jun 2003 - R. Yantosca - Initial Version
   (1 ) Now modified for GEOS-4 "a_llk_03" and "a_llk_04" fields (bmy, 3/22/04)
   (2 ) Modified for GCAP and GEOS-5 met fields (swu, bmy, 5/24/05)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
   27 Sep 2010 - R. Yantosca - Modified for start times other than 0 GMT
```

1.10.70 get_first_i6_time

Function GET_FIRST_I6_TIME returns the correct YYYYMMDD and HHMMSS values the first time that I-6 fields are read in from disk.

INTERFACE:

```
      FUNCTION GET_FIRST_I6_TIME() RESULT( DATE )
```

RETURN VALUE:

```
      INTEGER :: DATE(2)    ! YYYYMMDD, HHMMSS values
```

REVISION HISTORY:

```
   27 Sep 2010 - R. Yantosca - Initial version
```

1.10.71 its_time_for_chem

Function ITS_TIME_FOR_CHEM returns TRUE if it is time to do chemistry, or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_CHEM() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.72 its_time_for_conv

Function ITS_TIME_FOR_CONV returns TRUE if it is time to do convection, or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_CONV() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.73 its_time_for_dyn

Function ITS_TIME_FOR_DYN returns TRUE if it is time to do chemistry and false otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_DYN() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.74 its_time_for_emis

Function ITS_TIME_FOR_EMIS returns TRUE if it is time to do emissions, or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_EMIS() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.75 its_time_for_unit

Function ITS_TIME_FOR_UNIT returns TRUE if it is time to do unit conversion, or FALSE otherwise.

INTERFACE:

```
FUNCTION ITS_TIME_FOR_UNIT() RESULT( FLAG )
```

RETURN VALUE:

```
LOGICAL :: FLAG
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.76 its_time_for_diag

Function ITS_TIME_FOR_DIAG returns TRUE if it is time to archive certain diagnostics, or FALSE otherwise.

INTERFACE:

```
FUNCTION ITS_TIME_FOR_DIAG() RESULT( FLAG )
```

RETURN VALUE:

```
LOGICAL :: FLAG
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
20 Jul 2009 - C. Carouge  - Use TS_DIAG now and not 60 minutes
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.77 its_time_for_a1

Function ITS_TIME_FOR_A1 returns TRUE if it is time to read in A1 (average 1-hr fields) and FALSE otherwise.

INTERFACE:

```
FUNCTION ITS_TIME_FOR_A1() RESULT( FLAG )
```

RETURN VALUE:

```
LOGICAL :: FLAG
```

REVISION HISTORY:

```
20 Aug 2010 - R. Yantosca - Initial version
```

1.10.78 its_time_for_a3

Function ITS_TIME_FOR_A3 returns TRUE if it is time to read in A3 (average 3-hr fields) and FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_A3() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.79 its_time_for_a6

Function ITS_TIME_FOR_A6 returns TRUE if it is time to read in A6 (average 6-hr fields) and FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_A6() RESULT( FLAG )
```

USES:

```
 #     include "define.h"
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   (1 ) Now compute when it's time to read in GEOS-4 A-6 fields. (bmy, 6/26/03)
   (2 ) Now modified for GEOS-4 "a_llk_03" and "a_llk_04" fields (bmy, 3/22/04)
   (3 ) Now modified for GCAP and GEOS-5 met fields (swu, bmy, 5/24/05)
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.80 its_time_for_i6

Function ITS_TIME_FOR_I6 returns TRUE if it is time to read in I6 (instantaneous 6-hr fields) and FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_I6() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
  21 Mar 2003 - R. Yantosca - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.81 its_time_for_unzip

Function ITS_TIME_FOR_UNZIP Treturns TRUE if it is time to unzip the next day's met field files, or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_UNZIP() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
  21 Mar 2003 - R. Yantosca - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.82 its_time_for_del

Function ITS_TIME_FOR_DEL returns TRUE if it is time to delete the previous day's met field files in the temporary directory.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_DEL() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
  21 Mar 2003 - R. Yantosca - Initial Version
  19 Jun 2003 - R. Yantosca - Now delete files at 23 GMT each day, since the
                              last fvDAS A-3 field is 22:30 GMT and the last
                              fvDAS A-6 field is 21 GMT
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.83 its_time_for_exit

Function ITS_TIME_FOR_EXIT returns TRUE if it is the end of the GEOS-Chem simulation (i.e. TAU ¿= TAUe), or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_EXIT() RESULT( FLAG )
```

RETURN VALUE:

```
      LOGICAL :: FLAG
```

REVISION HISTORY:

```
    21 Mar 2003 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.84 its_time_for_bpch

Function ITS_TIME_FOR_BPCH returns TRUE if it's time to write output to the bpch file, or FALSE otherwise.

INTERFACE:

```
      FUNCTION ITS_TIME_FOR_BPCH() RESULT( DO_BPCH )
```

USES:

```
 #     include "CMN_SIZE"  ! Size parameters
 #     include "CMN_DIAG"  ! NJDAY
```

RETURN VALUE:

```
      LOGICAL :: DO_BPCH
```

REVISION HISTORY:

```
    02 Feb 2007 - R. Yantosca - Initial Version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.85 its_a_leapyear

Function ITS_A_LEAPYEAR tests to see if a year is really a leapyear.

INTERFACE:

```
      FUNCTION ITS_A_LEAPYEAR( YEAR_IN, FORCE ) RESULT( IS_LEAPYEAR )
```

INPUT PARAMETERS:

```
INTEGER, INTENT(IN), OPTIONAL :: YEAR_IN   ! Year to test if leapyear
LOGICAL, INTENT(IN), OPTIONAL :: FORCE     ! Do not exit if using GCAP
```

RETURN VALUE:

```
LOGICAL                       :: IS_LEAPYEAR  ! =T if it's a leapyear
```

REVISION HISTORY:

```
17 Mar 1999 - R. Yantosca - Initial Version
(1 ) Now remove YEAR from ARG list; use the module variable (bmy, 3/21/03)
(2 ) Now add YEAR_IN as an optional argument.  If YEAR_IN is not passed,
      then test if the current year is a leapyear (bmy, 9/25/03)
(3 ) Now always return FALSE for GCAP (swu, bmy, 8/29/05)
(4 ) Now add FORCE argument to force ITS_A_LEAPYEAR to return a value
      instead of just returning with FALSE for the GCAP met fields.
      (swu, bmy, 4/24/06)
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.86 its_a_new_year

Function ITS_A_NEW_YEAR returns TRUE if it's the first of a new month (it also returns TRUE on the first timestep of the run). This is useful for setting flags for reading in data.

INTERFACE:

```
FUNCTION ITS_A_NEW_YEAR() RESULT( IS_NEW_YEAR )
```

RETURN VALUE:

```
LOGICAL :: IS_NEW_YEAR
```

REVISION HISTORY:

```
01 Apr 2004 - R. Yantosca - Initial Version
01 Nov 2005 - R. Yantosca - Bug fix: Need month & day to be 1
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.87 its_a_new_month

Function ITS_A_NEW_MONTH returns TRUE if it's the first of a new month (it also returns TRUE on the first timestep of the run). This is useful for setting flags for reading in data.

INTERFACE:

```
      FUNCTION ITS_A_NEW_MONTH() RESULT( IS_NEW_MONTH )
```

RETURN VALUE:

```
      LOGICAL :: IS_NEW_MONTH
```

REVISION HISTORY:

```
  01 Apr 2004 - R. Yantosca - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.88 its_midmonth

Function ITS_MIDMONTH returns TRUE if it's the middle of a month.

INTERFACE:

```
      FUNCTION ITS_MIDMONTH() RESULT( IS_MIDMONTH )
```

RETURN VALUE:

```
      LOGICAL :: IS_MIDMONTH
```

REVISION HISTORY:

```
  10 Oct 2005 - S. Strode   - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.89 its_a_new_season

Function ITS_A_NEW_DAY returns TRUE if it's the first timestep of a new day (it also returns TRUE on the first timestep of the run). This is useful for setting flags for reading in data.

INTERFACE:

```
      FUNCTION ITS_A_NEW_DAY( ) RESULT( IS_NEW_DAY )
```

RETURN VALUE:

```
      LOGICAL :: IS_NEW_DAY
```

REVISION HISTORY:

```
  01 Apr 2004 - R. Yantosca - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.90 its_a_new_season

Function ITS_A_NEW_SEASON returns TRUE if it's a new season or FALSE if it's not a new season. Seasons are (1=DJF, 2=MAM, 3=JJA, 4=SON).

INTERFACE:

```
      FUNCTION ITS_A_NEW_SEASON( ) RESULT( IS_NEW_SEASON )
```

RETURN VALUE:

```
      LOGICAL :: IS_NEW_SEASON
```

REVISION HISTORY:

```
   20 Jul 2004 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.91 print_current_time

Subroutine PRINT_CURRENT_TIME prints the date, GMT time, and elapsed hours of a GEOS-Chem simulation.

INTERFACE:

```
      SUBROUTINE PRINT_CURRENT_TIME
```

REVISION HISTORY:

```
   21 Mar 2003 - R. Yantosca - Initial Version
   15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.92 timestamp_string

Function TIMESTAMP_STRING returns a formatted string "YYYY/MM/DD hh:mm" for the a date and time specified by YYYYMMDD and hhmmss. If YYYYMMDD and hhmmss are omitted, then TIMESTAMP_STRING will create a formatted string for the current date and time.

INTERFACE:

```
      FUNCTION TIMESTAMP_STRING( YYYYMMDD, HHMMSS ) RESULT( TIME_STR )
```

USES:

```
 #    include "define.h"
```

INPUT PARAMETERS:

```
    INTEGER, INTENT(IN), OPTIONAL :: YYYYMMDD   ! YYYY/MM/DD date
    INTEGER, INTENT(IN), OPTIONAL :: HHMMSS     ! hh:mm:ss time
```

RETURN VALUE:

```
    CHARACTER(LEN=16)             :: TIME_STR
```

REVISION HISTORY:

```
21 Mar 2003 - R. Yantosca - Initial Version
(1 ) Now use ENCODE statement for PGI/F90 on Linux (bmy, 9/29/03)
(2 ) Now add optional arguments YYYYMMDD and HHMMSS (bmy, 10/27/03)
(3 ) Renamed LINUX to LINUX_PGI (bmy, 12/2/03)
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.93 ymd_extract

Subroutine YMD_EXTRACT extracts the year, month, and date from an integer variable in YYYYMMDD format. It can also extract the hours, minutes, and seconds from a variable in HHMMSS format.

INTERFACE:

```
    SUBROUTINE YMD_EXTRACT( NYMD, Y, M, D )
```

INPUT PARAMETERS:

```
    INTEGER, INTENT(IN)  :: NYMD      ! YYYY/MM/DD format date
```

OUTPUT PARAMETERS:

```
    INTEGER, INTENT(OUT) :: Y, M, D   ! Separated YYYY, MM, DD values
```

REVISION HISTORY:

```
21 Nov 2001 - R. Yantosca - Initial Version
15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.94 expand_date

Subroutine EXPAND_DATE replaces "YYYYMMDD" and "hhmmss" tokens within a filename string with the actual values.

INTERFACE:

```
    SUBROUTINE EXPAND_DATE( FILENAME, YYYYMMDD, HHMMSS )
```

USES:

```
      USE CHARPAK_MOD, ONLY : STRREPL

#     include "define.h"
```

INPUT PARAMETERS:

```
      INTEGER,          INTENT(IN)    :: YYYYMMDD   ! YYYY/MM/DD date
      INTEGER,          INTENT(IN)    :: HHMMSS     ! hh:mm:ss time
```

INPUT/OUTPUT PARAMETERS:

```
      CHARACTER(LEN=*), INTENT(INOUT) :: FILENAME   ! Filename to modify
```

REVISION HISTORY:

```
  27 Jun 2002 - R. Yantosca - Initial Version
  (1 ) Bug fix for Linux: use ENCODE statement to convert number to string
        instead of F90 internal read. (bmy, 9/29/03)
  (2 ) Now replace 2 and 4 digit year strings for all models (bmy, 10/23/03)
  (3 ) Renamed LINUX to LINUX_PGI (bmy, 12/2/03)
  (4 ) Now do not replace "ss" with seconds, as the smallest GEOS-Chem
        timestep is in minutes. (bmy, 7/20/04)
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.95 system_date_time

Subroutine SYSTEM_DATE_TIME returns the actual local date and time (as opposed to the model date and time).

INTERFACE:

```
      SUBROUTINE SYSTEM_DATE_TIME( SYS_NYMD, SYS_NHMS )
```

OUTPUT PARAMETERS:

```
      INTEGER, INTENT(OUT) :: SYS_NYMD   ! System date in YYYY/MM/DD format
      INTEGER, INTENT(OUT) :: SYS_NHMS   ! System time in YYYY/MM/DD format
```

REMARKS:

```
  Uses the F90 intrinsic function DATE_AND_TIME.
```

REVISION HISTORY:

```
  02 May 2005 - R. Yantosca - Initial Version
  15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.96 system_timestamp

Function SYSTEM_TIMESTAMP returns a 16 character string with the system date and time in YYYY/MM/DD HH:MM format.

INTERFACE:

```
      FUNCTION SYSTEM_TIMESTAMP() RESULT( STAMP )
```

RETURN VALUE:

```
      CHARACTER(LEN=16) :: STAMP
```

REVISION HISTORY:

```
    03 May 2005 - R. Yantosca - Initial version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.97 timestamp_diag

Subroutine TIMESTAMP_DIAG save timestamps to be used in filenames for diagnostics. We do not want the time when the diagnostic is saved but the time for previous dynamic time step because midnight is considered as the beginning of next day (and not ending of previous day).

INTERFACE:

```
      SUBROUTINE TIMESTAMP_DIAG
```

REVISION HISTORY:

```
    12 Aug 2009 - C. Carouge  - Initial version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.10.98 get_nymd_diag

Function GET_NYMD_DIAG returns the previous NYMD value (YYYYMMDD) to the calling program. Used for diagnostic filenames.

INTERFACE:

```
      FUNCTION GET_NYMD_DIAG() RESULT( THISNYMD )
```

RETURN VALUE:

```
      INTEGER :: THISNYMD
```

REVISION HISTORY:

```
    12 Aug 2009 - C. Carouge  - Initial version
    15 Jan 2010 - R. Yantosca - Added ProTeX headers
```

1.11 Fortran: Module Interface transfer_mod.f

Module TRANSFER_MOD contains routines used to copy data from REAL*4 to REAL*8 arrays after being read from disk. Also, vertical levels will be collapsed in the stratosphere if necessary. This will help us to gain computational advantage.

INTERFACE:

```
      MODULE TRANSFER_MOD
```

USES:

```
      USE ERROR_MOD, ONLY : ALLOC_ERR
      USE ERROR_MOD, ONLY : GEOS_CHEM_STOP

      IMPLICIT NONE
      PRIVATE

#     include "CMN_SIZE"
```

PUBLIC MEMBER FUNCTIONS:

```
      PUBLIC  :: TRANSFER_A6
      PUBLIC  :: TRANSFER_2D
      PUBLIC  :: TRANSFER_3D
      PUBLIC  :: TRANSFER_3D_Lp1
      PUBLIC  :: TRANSFER_3D_TROP
      PUBLIC  :: TRANSFER_G5_PLE
      PUBLIC  :: TRANSFER_ZONAL
      PUBLIC  :: TRANSFER_TO_1D
      PUBLIC  :: INIT_TRANSFER
      PUBLIC  :: CLEANUP_TRANSFER

      INTERFACE TRANSFER_2D
         MODULE PROCEDURE TRANSFER_2D_INT
         MODULE PROCEDURE TRANSFER_2D_R4
         MODULE PROCEDURE TRANSFER_2D_R8
      END INTERFACE

      INTERFACE TRANSFER_ZONAL
         MODULE PROCEDURE TRANSFER_ZONAL_R4
         MODULE PROCEDURE TRANSFER_ZONAL_R8
      END INTERFACE
```

PRIVATE MEMBER FUNCTIONS:

```
      PRIVATE :: LUMP_2
      PRIVATE :: LUMP_2_R4
      PRIVATE :: LUMP_2_R8
      PRIVATE :: LUMP_4
      PRIVATE :: LUMP_4_R4
      PRIVATE :: LUMP_4_R8
      PRIVATE :: TRANSFER_2D_INT
      PRIVATE :: TRANSFER_2D_R4
      PRIVATE :: TRANSFER_2D_R8
      PRIVATE :: TRANSFER_ZONAL_R4
      PRIVATE :: TRANSFER_ZONAL_R8

      INTERFACE LUMP_2
         MODULE PROCEDURE LUMP_2_R4
         MODULE PROCEDURE LUMP_2_R8
      END INTERFACE

      INTERFACE LUMP_4
         MODULE PROCEDURE LUMP_4_R4
         MODULE PROCEDURE LUMP_4_R8
      END INTERFACE
```

REMARKS:

```
   Hybrid Grid Coordinate Definition: (dsa, bmy, 8/27/02, 8/13/10)
   ============================================================================
                                                                              .
   GEOS-4, GEOS-5, and MERRA (hybrid grids):
   ----------------------------------------------------------------------------
   For GEOS-4 and GEOS-5, the pressure at the bottom edge of grid box (I,J,L)
   is defined as follows:
                                                                              .
      Pedge(I,J,L) = Ap(L) + [ Bp(L) * Psurface(I,J) ]
                                                                              .
   where
                                                                              .
      Psurface(I,J) is  the "true" surface pressure at lon,lat (I,J)
      Ap(L)         has the same units as surface pressure [hPa]
      Bp(L)         is  a unitless constant given at level edges
                                                                              .
   Ap(L) and Bp(L) are given to us by GMAO.
                                                                              .
                                                                              .
   GEOS-3 (pure-sigma) and GCAP (hybrid grid):
   ----------------------------------------------------------------------------
   GEOS-3 is a pure-sigma grid.  GCAP is a hybrid grid, but its grid is
   defined as if it were a pure sigma grid (i.e. PTOP=150 hPa, and negative
   sigma edges at higher levels).  For these grids, can stil use the same
   formula as for GEOS-4, with one modification:
                                                                              .
      Pedge(I,J,L) = Ap(L) + [ Bp(L) * ( Psurface(I,J) - PTOP ) ]
                                                                              .
   where
```

```
                                                                           .
     Psurface(I,J) = the "true" surface pressure at lon,lat (I,J)
     Ap(L)         = PTOP    = model top pressure
     Bp(L)         = SIGE(L) = bottom sigma edge of level L
                                                                           .
                                                                           .
  The following are true for GCAP, GEOS-3, GEOS-4:
  ----------------------------------------------------------------------------
  (1) Bp(LLPAR+1) = 0.0          (L=LLPAR+1 is the atmosphere top)
  (2) Bp(1)       = 1.0          (L=1       is the surface       )
  (3) PTOP        = Ap(LLPAR+1)  (L=LLPAR+1 is the atmosphere top)
```

REVISION HISTORY:

```
  21 Sep 2010 - M. Evans    - Initial version
  (1 ) GEOS-3 Output levels were determined by Mat Evans.  Groups of 2 levels
        and groups of 4 levels on the original grid are merged together into
        thick levels for the output grid. (mje, bmy, 9/26/01)
  (2 ) Assumes that LLPAR == LGLOB for GEOS-1, GEOS-STRAT (bmy, 9/26/01)
  (3 ) EDGE_IN needs to be provided for each model type, within an #ifdef
        block, in order to ensure compilation.  However, EDGE_IN is currently
        only used for regridding GEOS-3 data (and probably also GEOS-4 when
        that becomes available). (bmy, 9/26/01)
  (4 ) Add interfaces TRANSFER_2D and TRANSFER_ZONAL (bmy, 9/27/01)
  (5 ) Added routine TRANSFER_2D_R4.  Added TRANSFER_2D_R4 to the generic
        TRANSFER_2D interface. (bmy, 1/25/02)
  (6 ) Updated comments, cosmetic changes (bmy, 2/28/02)
  (7 ) Bug fix: remove extraneous "," in GEOS-1 definition of EDGE_IN array.
        (bmy, 3/25/02)
  (8 ) Now divide module header into MODULE PRIVATE, MODULE VARIABLES, and
        MODULE ROUTINES sections.  Also add MODULE INTERFACES section,
        since we have an interface here. (bmy, 5/28/02)
  (9 ) Now references "pressure_mod.f" (dsa, bdf, bmy, 8/22/02)
  (10) Bug fix in "init_transfer", declare variable L.  Also reference
        GEOS_CHEM_STOP from "error_mod.f" for safe stop (bmy, 10/15/02)
  (11) Added routine TRANSFER_3D_TROP.  Also updated comments. (bmy, 10/31/02)
  (12) Now uses functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f".
        (bmy, 3/11/03)
  (13) Added code to regrid GEOS-4 from 55 --> 30 levels.  Renamed module
        variable SIGE_IN to EDGE_IN. (mje, bmy, 10/31/03)
  (14) Now modified for GEOS-5 and GCAP met fields (swu, bmy, 5/24/05)
  (15) Remove support for GEOS-1 and GEOS-STRAT met fields (bmy, 8/4/06)
  (16) Modified for GEOS-5.  Rewritten for clarity. (bmy, 10/30/07)
  13 Aug 2010 - R. Yantosca - Added modifications for MERRA met fields
  13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.1 transfer_a6

Subroutine TRANSFER_A6 transfers A-6 data from a REAL*4 array to a REAL*8 array. Vertical layers are collapsed (from LGLOB to LLPAR) if necessary.

INTERFACE:

```
      SUBROUTINE TRANSFER_A6( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB,LGLOB)     ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(LLPAR,IIPAR,JJPAR)    ! Output data
```

REMARKS:

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) A-6 fields are dimensioned (LLPAR,IIPAR,JJPAR) since for Fortran
         efficiency, since the code loops over vertical layers L in a column
         located above a certain surface location (I,J). (bmy, 9/21/01)
   (2 ) Assumes that LLPAR == LGLOB for GEOS-1, GEOS-STRAT (bmy, 9/21/01)
   (3 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f".
         Now I0, J0 are local variables. (bmy, 3/11/03)
   (4 ) Added code to regrid GEOS-4 from 55 --> 30 levels (mje, bmy, 10/31/03)
   (5 ) Now modified for GEOS-5 met fields (bmy, 5/24/05)
   (6 ) Rewritten for clarity (bmy, 2/8/07)
   (7 ) Now get nested-grid offsets (dan, bmy, 11/6/08)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
   13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                               the vertical grids are identical
```

1.11.2

Subroutine TRANSFER_3D transfers 3-dimensional data from a REAL*4 array to a REAL*8 array. Vertical layers are collapsed (from LGLOB to LLPAR) if necessary.

INTERFACE:

```
      SUBROUTINE TRANSFER_3D( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IIPAR,JJPAR,LGLOB)    ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(IIPAR,JJPAR,LLPAR)   ! Output data
```

REVISION HISTORY:

```
  19 Sep 2001 - R. Yantosca - Initial version
  (1 ) Lump levels together in groups of 2 or 4, as dictated by Mat Evans.
        (bmy, 9/21/01)
  (2 ) Assumes that LLPAR == LGLOB for GEOS-1, GEOS-STRAT (bmy, 9/21/01)
  (3 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f".
        Now I0, J0 are local variables. (bmy, 3/11/03)
  (4 ) Added code to regrid GEOS-4 from 55 --> 30 levels (mje, bmy, 10/31/03)
  (5 ) Now modified for GEOS-5 met fields (bmy, 5/24/05)
  (6 ) Rewritten for clarity (bmy, 2/8/07)
  13 Aug 2010 - R. Yantosca - Added ProTeX headers
  13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                              the vertical grids are identical
```

1.11.3 transfer_g5_ple

Subroutine TRANSFER_G5_PLE transfers GEOS-5/MERRA pressure edge data from the native 72-level grid to the reduced 47-level grid.

INTERFACE:

```
      SUBROUTINE TRANSFER_G5_PLE( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IIPAR,JJPAR,LGLOB+1)    ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(IIPAR,JJPAR,LLPAR+1)   ! Output data
```

REVISION HISTORY:

```
  08 Feb 2007 - R. Yantosca - Initial version
  13 Aug 2010 - R. Yantosca - Added ProTeX headers
  13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                              the vertical grids are identical
```

1.11.4 transfer_3d_lp1

Subroutine TRANSFER_3D_Lp1 transfers 3-D data from a REAL*4 array of dimension (IGLOB,JGLOB,LGLOB+1) to a REAL*8 array of dimension (IIPAR,JJPAR,LLPAR+1). Regrid in the vertical if needed.

INTERFACE:

```
      SUBROUTINE TRANSFER_3D_Lp1( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IIPAR,JJPAR,LGLOB+1)    ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(IIPAR,JJPAR,LLPAR+1)   ! Output data
```

REVISION HISTORY:

```
  08 Feb 2007 - R. Yantosca - Initial version
  13 Aug 2010 - R. Yantosca - Added ProTeX headers
  13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                              the vertical grids are identical
```

1.11.5 transfer_3d_trop

Subroutine TRANSFER_3D_TROP transfers tropospheric 3-D data from a REAL*4 array to a REAL*8 array.

INTERFACE:

```
      SUBROUTINE TRANSFER_3D_TROP( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB,LLTROP_FIX)     ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(IIPAR,JJPAR,LLTROP_FIX)    ! Output data
```

REVISION HISTORY:

```
  19 Sep 2001 - M. Evans    - Initial version
  08 Feb 2007 - R. Yantosca - Now use LLTROP_FIX instead of LLTROP, since
                              most of the offline simulations use the annual
                              mean tropopause
  13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.6 transfer_zonal_r4

Subroutine TRANSFER_ZONAL_R4 transfers zonal-mean data from a REAL*4 array to a REAL*8 array. Vertical levels are collapsed (from LGLOB to LLPAR) if necessary. (mje, bmy, 9/21/01, 2/8/07)

INTERFACE:

```
      SUBROUTINE TRANSFER_ZONAL_R4( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(JGLOB,LGLOB)     ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*4,  INTENT(OUT) :: OUT(JJPAR,LLPAR)    ! Output data
```

REVISION HISTORY:

```
   19 Sep 2001 - M. Evans    - Initial version
   (1 ) Lump levels together in groups of 2 or 4, as dictated by Mat Evans.
         (bmy, 9/21/01)
   (2 ) Assumes that LLPAR == LGLOB for GEOS-1, GEOS-STRAT (bmy, 9/21/01)
   (3 ) Now use function GET_YOFFSET from "grid_mod.f".  Now I0 and J0 are
         local variables (bmy, 3/11/03)
   (4 ) Added code to regrid GEOS-4 from 55 --> 30 levels (mje, bmy, 10/31/03)
   (5 ) Rewritten for clarity (bmy, 2/8/07)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
   13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                               the vertical grids are identical
```

1.11.7 transfer_zonal_r8

Subroutine TRANSFER_ZONAL_R8 transfers zonal mean or lat-alt data from a REAL*4 array of dimension (JGLOB,LGLOB) to a REAL*8 array of dimension (JJPAR,LLPAR). Regrid data in the vertical if necessary by lumping levels.

INTERFACE:

```
      SUBROUTINE TRANSFER_ZONAL_R8( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(JGLOB,LGLOB)    ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(JJPAR,LLPAR)   ! Output data
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Lump levels together in groups of 2 or 4, as dictated by Mat Evans.
         (bmy, 9/21/01)
   (2 ) Assumes that LLPAR == LGLOB for GEOS-1, GEOS-STRAT (bmy, 9/21/01)
   (3 ) Now use function GET_YOFFSET from "grid_mod.f".  Now I0 and J0 are
         local variables (bmy, 3/11/03)
```

```
    (4 ) Added code to regrid GEOS-4 from 55 --> 30 levels (mje, bmy, 10/31/03)
    (5 ) Now modified for GEOS-5 met fields (bmy, 5/24/05)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
    13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                                the vertical grids are identical
```

1.11.8 transfer_2d_int

Subroutine TRANSFER_2D_INT transfers 2-D data from a REAL*4 array of dimension (IGLOB,JGLOB) to an INTEGER array of dimension (IIPAR,JJPAR).

INTERFACE:

```
      SUBROUTINE TRANSFER_2D_INT( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB)    ! Input data
```

OUTPUT PARAMETERS:

```
      INTEGER, INTENT(OUT) :: OUT(IIPAR,JJPAR)   ! Output data
```

REVISION HISTORY:

```
    19 Sep 2001 - R. Yantosca - Initial version
    (1 ) Use parallel DO loops to speed things up (bmy, 9/21/01)!
    (2 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f".
         Now I0 and J0 are local variables. (bmy, 3/11/03)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.9 transfer_2d_r4

Subroutine TRANSFER_2D_R4 transfers 2-D data from a REAL*4 array of dimension (IGLOB,JGLOB) to a REAL*4 array of dimension (IIPAR,JJPAR).

INTERFACE:

```
      SUBROUTINE TRANSFER_2D_R4( IN, OUT )
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB)
```

OUTPUT PARAMETERS:

```
      REAL*4,  INTENT(OUT) :: OUT(IIPAR,JJPAR)
```

REVISION HISTORY:

```
    19 Sep 2001 - R. Yantosca - Initial version
    (1 ) Use parallel DO loops to speed things up (bmy, 9/21/01)
    (2 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f"
          Now I0 and J0 are local variables (bmy, 3/11/03)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.10 transfer_2d_r8

Subroutine TRANSFER_2D_R8 transfers 2-D data from a REAL*4 array of dimension (IGLOB,JGLOB) to a REAL*8 array of dimension (IIPAR,JJPAR).

INTERFACE:

```
       SUBROUTINE TRANSFER_2D_R8( IN, OUT )
```

INPUT PARAMETERS:

```
       REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB)    ! Input data
```

OUTPUT PARAMETERS:

```
       REAL*8,  INTENT(OUT) :: OUT(IIPAR,JJPAR)   ! Output data
```

REVISION HISTORY:

```
    19 Sep 2001 - R. Yantosca - Initial version
    (1 ) Use parallel DO loops to speed things up (bmy, 9/21/01)
    (2 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f"
          Now I0 and J0 are local variables. (bmy, 3/11/03)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.11 transfer_to_1d

Subroutine TRANSFER_TO_1D transfers 2-D data from a REAL*4 array of dimension (IGLOB,JGLOB) to 1-D a REAL*8 array of dimension (MAXIJ), where MAXIJ = IIPAR * JJPAR.

INTERFACE:

```
       SUBROUTINE TRANSFER_TO_1D( IN, OUT )
```

INPUT PARAMETERS:

```
       REAL*4,  INTENT(IN)  :: IN(IGLOB,JGLOB)    ! Input data
```

OUTPUT PARAMETERS:

```
      REAL*8,  INTENT(OUT) :: OUT(MAXIJ)         ! Output data
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Use single-processor DO-loops for now (bmy, 9/21/01)
   (2 ) Now use functions GET_XOFFSET and GET_YOFFSET from "grid_mod.f".
         Now I0 and J0 are local variables. (bmy, 3/11/03)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.12 lump_2_r4

Function LUMP_2_R4 lumps 2 sigma levels into one thick level. Input arguments must be REAL*4.

INTERFACE:

```
      FUNCTION LUMP_2_R4( IN, L_IN, L ) RESULT( OUT )
```

USES:

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN) :: IN(L_IN)   ! Column of data on input grid
      INTEGER, INTENT(IN) :: L_IN       ! Vertical dimension of the IN array
      INTEGER, INTENT(IN) :: L          ! Level on input grid from which
                                        !  to start regridding
```

RETURN VALUE:

```
      REAL*4             :: OUT        ! Data on output grid: 4 lumped levels
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Now references GEOS_CHEM_STOP from "error_mod.f" (bmy, 10/15/02)
   (2 ) Renamed SIGE_IN to EDGE_IN to denote that it is not always a sigma
         coordinate (as for GEOS-4).  Also updated comments (bmy, 10/31/03)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.13 lump_2_r8

Function LUMP_2_R8 lumps 2 sigma levels into one thick level. Input arguments must be REAL*8.

INTERFACE:

```
      FUNCTION LUMP_2_R8( IN, L_IN, L ) RESULT( OUT )
```

USES:

```
      USE ERROR_MOD, ONLY : GEOS_CHEM_STOP
```

INPUT PARAMETERS:

```
      REAL*8,  INTENT(IN) :: IN(L_IN)   ! Column of data on input grid
      INTEGER, INTENT(IN) :: L_IN       ! Vertical dimension of the IN array
      INTEGER, INTENT(IN) :: L          ! Level on input grid from which
                                        !  to start regridding
```

RETURN VALUE:

```
      REAL*8              :: OUT        ! Data on output grid: 2 lumped levels
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Now references GEOS_CHEM_STOP from "error_mod.f" (bmy, 10/15/02)
   (2 ) Renamed SIGE_IN to EDGE_IN to denote that it is not always a sigma
         coordinate (as for GEOS-4).  Also updated comments (bmy, 10/31/03)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.14 lump_4_r4

Function LUMP_4_R4 lumps 4 sigma levels into one thick level. Input arguments must be REAL*4.

INTERFACE:

```
      FUNCTION LUMP_4_R4( IN, L_IN, L ) RESULT( OUT )
```

USES:

```
      USE ERROR_MOD, ONLY : GEOS_CHEM_STOP
```

INPUT PARAMETERS:

```
      REAL*4,  INTENT(IN) :: IN(L_IN)   ! Column of data on input grid
      INTEGER, INTENT(IN) :: L_IN       ! Vertical dimension of the IN array
      INTEGER, INTENT(IN) :: L          ! Level on input grid from which
                                        !  to start regridding
```

RETURN VALUE:

```
      REAL*4              :: OUT        ! Data on output grid: 4 lumped levels
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Now references GEOS_CHEM_STOP from "error_mod.f" (bmy, 10/15/02)
   (2 ) Renamed SIGE_IN to EDGE_IN to denote that it is not always a sigma
         coordinate (as for GEOS-4).  Also updated comments (bmy, 10/31/03)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.15 lump_4_r8

Function LUMP_4_R8 lumps 4 sigma levels into one thick level. Input arguments must be REAL*8.

INTERFACE:

```
      FUNCTION LUMP_4_R8( IN, L_IN, L ) RESULT( OUT )
```

USES:

```
      USE ERROR_MOD, ONLY : GEOS_CHEM_STOP
```

INPUT PARAMETERS:

```
      REAL*8,  INTENT(IN) :: IN(L_IN)   ! Column of data on input grid
      INTEGER, INTENT(IN) :: L_IN       ! Vertical dimension of the IN array
      INTEGER, INTENT(IN) :: L          ! Level on input grid from which
                                        !  to start regridding
```

RETURN VALUE:

```
      REAL*8              :: OUT        ! Data on output grid: 4 lumped levels
```

REVISION HISTORY:

```
   19 Sep 2001 - R. Yantosca - Initial version
   (1 ) Now references GEOS_CHEM_STOP from "error_mod.f" (bmy, 10/15/02)
   (2 ) Renamed SIGE_IN to EDGE_IN to denote that it is not always a sigma
         coordinate (as for GEOS-4).  Also updated comments (bmy, 10/31/03)
   13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.11.16 init_transfer

Subroutine INIT_TRANSFER initializes and zeroes all module variables.

INTERFACE:

```
      SUBROUTINE INIT_TRANSFER( THIS_I0, THIS_J0 )
```

USES:

INPUT PARAMETERS:

```
      INTEGER, INTENT(IN) :: THIS_I0    ! Global X (longitude) offset
      INTEGER, INTENT(IN) :: THIS_J0    ! Global Y (latitude)  offset
```

REVISION HISTORY:

```
    19 Sep 2001 - R. Yantosca - Initial version
    (1 ) Removed additional "," for GEOS-1 definition of EDGE_IN (bmy, 3/25/02)
    (2 ) Now use GET_BP from "pressure_mod.f" to get sigma edges for all
          grids except GEOS-3 (dsa, bdf, bmy, 8/22/02)
    (3 ) Declare L as a local variable.  Also reference ALLOC_ERR from module
          "error_mod.f" (bmy, 10/15/02)
    (4 ) Renamed SIGE_IN to EDGE_IN to denote that it is not always a sigma
          coordinate (as for GEOS-4).  Now assign original Ap coordinates from
          the GEOS-4 grid to the EDGE_IN array (bmy, 10/31/03)
    (5 ) Now modified for GEOS-5 met fields (bmy, 5/24/05)
    (6 ) Rewritten for clarity.  Remove references to "grid_mod.f" and
          "pressure_mod.f".  Now pass I0, J0 from "grid_mod.f" via the arg list.
           (bmy, 2/8/07)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
    13 Aug 2010 - R. Yantosca - Treat MERRA the same way as GEOS-5, because
                                the vertical grids are identical
```

1.11.17 cleanup_transfer

Subroutine CLEANUP_TRANSFER deallocates all module variables.

INTERFACE:

```
      SUBROUTINE CLEANUP_TRANSFER
```

REVISION HISTORY:

```
    19 Sep 2001 - R. Yantosca - Initial version
    31 Oct 2003 - R. Yantosca - Renamed SIGE_IN to EDGE_IN to denote that it
                                is not always a sigma coordinate (as for GEOS-4)
    13 Aug 2010 - R. Yantosca - Added ProTeX headers
```

1.12 Fortran: Module Interface unix_cmds_mod.f

Module UNIX_CMDS_MOD contains variables which contain file suffixes and various Unix command strings.

INTERFACE:

```
      MODULE UNIX_CMDS_MOD
```

USES:

```
      IMPLICIT NONE
      PUBLIC
```

PUBLIC DATA MEMBERS:

```
! Unix cmd and file suffix strings for ...
CHARACTER(LEN=255) :: BACKGROUND   ! Background operator  ( ' &'    )
CHARACTER(LEN=255) :: REDIRECT     ! Redirection operator ( ' >'    )
CHARACTER(LEN=255) :: REMOVE_CMD   ! File/dir remove cmd  ( 'rm'    )
CHARACTER(LEN=255) :: SEPARATOR    ! Dir path separator   ( '/'     )
CHARACTER(LEN=255) :: SPACE        ! Blank space          ( ' '     )
CHARACTER(LEN=255) :: UNZIP_CMD    ! Unzip command        ( 'gzcat' )
CHARACTER(LEN=255) :: WILD_CARD    ! Wild card operator   ( '*'     )
CHARACTER(LEN=255) :: A3_SUFFIX    ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: A6_SUFFIX    ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: I6_SUFFIX    ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: PH_SUFFIX    ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: KZZ_SUFFIX   ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: GRID_SUFFIX  ! !%%% OBSOLETE %%%
CHARACTER(LEN=255) :: ZIP_SUFFIX   ! Zipped file suffix   ( '.gz'   )
```

REVISION HISTORY:

```
09 Jul 2004 - R. Yantosca - Initial version
20 Nov 2009 - R. Yantosca - Added ProTeX header
```