CERES Software Bulletin 95-15

Toolkit SMF wrapper and toolkit.a Access

Alice Fan(t.f.fan@larc.nasa.gov)
Joe Stassi (j.c.stassi@larc.nasa.gov)

1.0 Purpose

This bulletin describes the Fortran 90 wrapper to the ECS Toolkit Status Message Facility, SMF, the lowest level Toolkit Tool. The SMF f90 wrapper is located in a CERESlib library file called toolkit.a. Wrappers to the Toolkit PCF and IO functions are also in CERESlib, but details about those wrappers are not included here. They will be addressed in a later bulletin.

2.0 What is the MSG module?

The MSG module is the Fortran 90 interface to SMF. This module is the lowest level module among all the wrapper modules--all other modules call it. It provides routines for writing error and status messages to log files. It also provides a means for flagging log and data files to be sent to users through ftp (with e-mail notification) when processing is complete.

3.0 Accessing the toolkit.a Library

The f90 MSG module is located in the CERESlib library file, toolkit.a, which resides under the $CERESLIB directory. Source code for the module can be found in the $CERESLIB/Source directory. Please refer to bulletin 95-13 for details on how to access CERESlib library files and how to create CERESlib'able Makefiles using the makemake utility. A sample Makefile generated by makemake is found in Appendix A. It illustrates the compilation flags and library links necessary for accessing CERESlib.

4.0 MSG Wrapper Routines

The MSG module makes the following PUBLIC routines available to the user:

(1) WriteReport and WriteStatus

(2) GetMsgByCode

(3) SendRunTimeData

A description of these routines follows in Sections 4.1 through 4.3.

4.1 WriteReport and WriteStatus: Message Output Routines

The MSG module contains two wrapper routines for outputting messages: WriteReport and WriteStatus. The WriteReport routine outputs a formatted message to the REPORT log file. The WriteStatus routine outputs a similarly formatted message to the STATUS log file and selected messages to the USER log file (see selection criteria in the mnemonic parameter description below).

The output messages for WriteReport and WriteStatus are determined by the input parameters: a mnemonic code, the calling routine name, a character string, and up to eight numeric values (four integer and four real). The input parameters are the same for both routines. They are ordered as follows:

Subroutine WriteReport (mnemonic, FuncName, StrVar, IVar1, IVar2, IVar3, IVar4, &
RVar1, RVar2, RVar3, RVar4)

Subroutine WriteStatus (mnemonic, FuncName, StrVar, IVar1, IVar2, IVar3, IVar4, &
RVar1, RVar2, RVar3, RVar4)

(1) mnemonic: This parameter is a mnemonic label from the Message Include File. It is assigned a system generated error number which points to a longer message string stored in Message Runtime File. For a detailed explanation of mnemonic labels and how to create Message files, refer to CERES_Bulletin 95-14. A listing of mnemonic severity levels is provided in Appendix B.

a) The mnemonic parameter is required by WriteStatus, but is optional for WriteReport.

b) WriteReport writes all messages to the REPORT log file. WriteStatus writes _N_ and _U_ level messages to both the STATUS and the USER log files, and other messages to the STATUS log file only.

c) If the severity level of the mnemonic label is _E_ or _F_, or if it is indeterminate, then the program will terminate.

(2) FuncName: This is the name of the calling routine, which could be a function or a subroutine. This parameter is optional, but users are strongly encouraged to supply it. If supplied, FuncName appears at the beginning of the output line, followed by a ":".

(3) StrVar: This parameter contains a string which gets printed. The string can either be static or dynamic. A dynamic string will contain integer tokens, %i, and real tokens, %f, which will be filled in order from the integer and real numeric parameters. This is an optional parameter.

(4) Numeric Parameters (IVar1, IVar2, IVar3, IVar4, RVar1, RVar2, RVar3, RVar4): These are integer (IVar's) and real (Rvar's) variables or numeric values. They are substituted into the integer and real tokens in the StrVar parameter. If StrVar does not exist or if it is not dynamic, or if there are more numeric parameters than tokens, then the leftover numeric values are appended to the end of the message. These parameters are optional.

Note that all the individual parameters are optional to the WriteReport routine. However, some parameters must be supplied in order to specify what gets outputted. Sample programs using the output routines, along with their outputs, are given in Section 5.0.

It is a good practice to remove the log files before each executable. Otherwise, messages are appended to the log files from previous run, which might cause confusion.

4.2 GetMsgByCode Routine

This subroutine retrieves the message string, CERES severity level, and the PGS severity level from the mnemonic code. The calling parameters for GetMsgByCode are ordered as follows:

Call GetMsgByCode (mnemonic, message, cereslevel, pgslevel)

The only input parameter is the mnemonic code explained above.

The output parameters are:

(1) message: Error message associated with the mnemonic code. This is a string which could be up to 240 character long.

(2) cereslevel: CERES severity level (0=Success, 1=Fatal, 2=Warning) corresponding to the PGS severity level.

(3) pgslevel: PGS severity level (512=Success, 1536=Message, 2048=User Information, 2560=Notice, 3072=Warning, 3584=Error, 4096=Fatal) indicated in the mnemonic label.

See Appendix B for a mapping between the PGS and CERES severity levels.

4.3 SendRunTimeData Routine

This subroutine provides the user with a method for flagging specific runtime data files for subsequent post-processing retrieval. The calling parameters are ordered as follows:

Call SendRunTimeData(NumberOfFiles, LogIdArray, VersionArray, Status)

The three input parameters are:

(1) NumberOfFiles: This is an integer value indicating the total number of files to be sent.

(2) LogIDArray: This array holds the logical file identifiers in the PCF file.

(3) VersionArray: This array holds the version numbers for the files identified in the preceding array of logical identifiers.

The only output parameter is Status which indicates the success or failure of the data file flagging procedure. The procedure will fail if the specified logical ids and the corresponding version numbers cannot be found in the Process Control File.

The marked files are transferred at the normal termination of a PGE process.

5.0 Message Output Examples

5.1 WriteReport

5.1.1 Calling Examples for the WriteReport Subroutine

```
 PROGRAM test_Reportlog
!*************************************************************************
! A driver to test the SMF Toolkit interface - WriteReport
!*************************************************************************
USE MSG, ONLY : WriteReport
IMPLICIT NONE
INCLUDE 'PGS_CERES_25000.f'
```

```
INTEGER status
CHARACTER(len=100) Message

call WriteReport(StrVar = "ERROR - The SFC file is not open - Status = ", &
                    IVar1 = 99)
call WriteReport(FuncName = "test_Reportlog()", &
                & StrVar= " The bounding rectangle are :” // &
                & ”( %f4.1, %f4.1, %f4.1, %f4.1 ) ",RVar1=3.1, RVar2=3.2, RVar3=3.3,
                 RVar4=3.4)
!
! No token is given, the Rvars will be appended at the end of the StrVar string
! with default format
!
call WriteReport(StrVar=" The bounding rectangle are : ",                    &
              RVar1=3.1, RVar2=3.2, RVar3=3.3, RVar4=3.4)

call WriteReport(StrVar = " Number of scan lines = %i3, number of Pixels=%i9." , &
                  IVar1 =64, Ivar2= 409)

call WriteReport(StrVar = " Degree = %f5.1", RVar1 = 55.5)

! Program is terminated when a message with level of _F_ or _E_ is written
!
  Message = " -- While open CID file"
  call writeReport(CERES_F_OPEN_MODE_ERROR, FuncName=" test_Reportlog()", &
                    StrVar= message )
END PROGRAM test_Reportlog
```

5.1.2 Outputs for the WriteReport Test Program

```
Thu Oct 12 14:34:31 1995
ERROR - The SFC file is not open - Status = 99
============================================================
Thu Oct 12 14:34:31 1995
test_Reportlog(): The bounding rectangle are :(  3.1,  3.2,  3.3,  3.4 )
```

```
==========================================================
Thu Oct 12 14:34:31 1995
 The bounding rectangle are :   3.0999999   3.2000000   3.3000000   3.4000001
==========================================================
Thu Oct 12 14:34:31 1995
 Number of scan lines =  64,  number of Pixels=       409.
==========================================================
Thu Oct 12 14:34:31 1995
 Degree =  55.5
==========================================================
Thu Oct 12 14:34:31 1995
 test_Reportlog():FATAL_ERROR...File open mode ERROR -- While open CID file
==========================================================
```

5.2 WriteStatus

The Message Text File, containing the mnemonic labels and message strings for the following examples, can be found in Appendix C.

5.2.1 Calling Examples for WriteStatus Subroutine

```
PROGRAM test_status
!*****************************************************************
! A driver to test the SMF Toolkit interface - WriteStatus
!*****************************************************************
USE  MSG, ONLY : WriteStatus
IMPLICIT NONE
INCLUDE 'PGS_CERES_25000.f'

INTEGER stat
CHARACTER(len=1) mode
CHARACTER(len=100) StrVar

call WriteStatus(CERES_M_BOUNDING_RECTANGLE,  &
                 StrVar=" (%f4.1,  %f4.1, %f4.1, %f4.1 ) ",RVar1=3.1, RVar2=3.2, &
                 RVar3=3.3, RVar4=3.4, FuncName="test_status()")
! incorrect format %4.1
call WriteStatus(CERES_M_BOUNDING_RECTANGLE,  StrVar=" ( %f4.1,  " // &
```

```
                "%4.1 %f4.1 %f4.1 ) ", RVar1=3.1, RVar2=3.2, RVar3=3.3, RVar4=3.4)
! If no token % is specified, all numbers are appended at the end
call WriteStatus(CERES_M_SENDDATA, FuncName="test_status()",  &
                IVar1 = 5)
stat = 35
call WriteStatus (CERES_M_MESSAGE, funame="test_status()",  &
                 StrVar="The first number is %i, the 2ND number is %f8.2,"//  &
                  & "The 3RD number is %f8.3", IVar1=stat, RVar1= 45.6, RVar2 =80.0)
!
! default format when %i or %f is used
!
call WriteStatus (CERES_M_MESSAGE, funcName="test_status()", &
                  StrVar= "The first number is %i, the 2ND number is  %f "// &
                 &"The 3RD number is %f",IVar1=stat, RVar1= 45.6, RVar2 =80.0)
!
! The following two messages are sent not only to the STATUS log but also the USER log
file
!
call WriteStatus(CERES_N_OPEN_MODE_ERROR, funcname="test_status()" )
!
call WriteStatus(CERES_U_MESSAGE, StrVar = " Degree = %f5.1", RVar1 = 55.5)

! To test a bad mnemonic number, which means the mnemonic number does not exist
! in any error message file. This stops the program in current approach.
! We might want to change it to a nonfatal error later.
!
 call WriteStatus(99, funcname="test_status()" )

!  Try to write a message with error level = "F" or "E" will terminate the program
mode = "W"
StrVar = " While open CID file with mode = "//mode
call WriteStatus(CERES_F_OPEN_MODE_ERROR, "test_status()", StrVar)

END PROGRAM test_status
```

5.2.2 Outputs for the WriteStatus Test Program

```
test_status():CERES_M_BOUNDING_RECTANGLE:204801559
The bounding rectangle = (  3.1,  3.2,  3.3,  3.4 )

Parse():CERES_W_TOKEN_FORMAT:204803094
ERROR TOKEN FORMAT, While try to write :The bounding rectangle =
( %f4.1, %4.1 %f4.1 %f4.1 )

test_status():CERES_M_SENDDATA:204801560
Message... Total number of files marked for sending: 5

test_status():CERES_M_MESSAGE:204801561
Message...The first number is  35, the 2ND number is    45.60, The 3RD number
is   80.000

test_status():CERES_M_MESSAGE:204801561
Message...The first number is  35, the 2ND number is   45.5999985 The 3RD number
 is   80.0000000

test_status():CERES_N_OPEN_MODE_ERROR:204802577
File mode error, use default - READ

():CERES_U_MESSAGE:204802074
User Information Degree =  55.5

PGetMsgByCode():PGSSMF_E_UNDEFINED_CODE:19968
 - caused by bad mnemonic code :  99

test_status():CERES_F_OPEN_MODE_ERROR:204804112
FATAL_ERROR...File open mode ERROR While open CID file with mode = W

FINUTL():CERES_M_ACTION_ERROR:204801537

================================================
```

**********Outputs In the LogStatus file

test_Userlog():CERES_N_OPEN_MODE_ERROR:204802577
File mode error, use default - READ
():CERES_U_MESSAGE:204802074
User Information Degree = 55.5

Appendix A: Sample Makefile to Link with CERESlib

```
PROG = test_status.exe

VPATH = $(CERESLIB)/Include:../Include:./Include

SRCS = test_status.f90

OBJS = test_status.o
OBJS_F =

CERESMOD_FLAG = -I$(CERESLIB)/Mod
CERESINC_FLAG = -I$(CERESLIB)/Include
OTHERINC_FLAG = -I../Include  -I./Include

PGSHOME = /opt/net/PGSTKV5
PGSLIB = $(PGSHOME)/lib

CLIBS = $(CERESLIB)/data_products.a $(CERESLIB)/toolkit.a \
            $(CERESLIB)/cereslib.a
TKLIBS = -L$(PGSLIB) -lPGSTK -lansi
LIBS =

CC = acc
CFLAGS = -O
FC = f77
FFLAGS = -O
F90 = f90
F90FLAGS = -O $(CERESMOD_FLAG) $(CERESINC_FLAG) $(OTHERINC_FLAG)
LDFLAGS = -s

all: $(PROG)

$(PROG): $(OBJS) $(OBJS_F) $(CLIBS)
            $(F90) $(LDFLAGS) -o $@ $(OBJS) $(OBJS_F) $(LIBS) $(CLIBS)
$(TKLIBS)

clean:
            rm -f $(PROG) $(OBJS) $(OBJS_F) *.mod

.SUFFIXES:

.SUFFIXES: .f90 .mod .c .o

.f90.o:
            $(F90) $(F90FLAGS) -c $<

.f90.mod:
            $(F90) $(F90FLAGS) -c $<

.c.o:
            $(CC) $(CFLAGS) -c $<

test_status.o: PGS_CERES_25000.
```

Appendix B: PGS and CERES Severity Levels

Seven PGS levels map into 3 CERES levels.

mnemonic code	PGS level	CERES level
_S_	512 (= Success)	0 (= Success)
_M_	1536 (= Message)	"
_U_	2048 (= User Information)	"
_N_	2560 (= Notice)	"
_W_	3072 (= Warning)	2 (= Warning)
_E_	3584 (= Error)	1 (= Fatal)
_F_	4096 (= Fatal)	"

Appendix C: Message Text File for Examples in Section 5.0.

```
%INSTR  = CERES
%LABEL  = CERES
%SEED   = 25000
#
CERES_M_STOP_NORMAL   STOP NORMAL
CERES_M_ACTION_ERROR    TERMINATED BY MESSAGE ERROR LEVEL (ACTION= -1)
#
CERES_F_OPEN_MODE_ERROR   FATAL_ERROR...File open mode ERROR
CERES_N_OPEN_MODE_ERROR   File mode error, use default - READ
#
CERES_W_TOKEN_FORMAT ERROR...Token Format is incorrect
CERES_M_BOUNDING_RECTANGLE The bounding rectangle =
CERES_M_SENDDATA  Message... Total number of files marked for sending:
CERES_M_MESSAGE Message...
CERES_U_MESSAGE User Information
CERES_F_OPEN_BINARY_FILE  FATAL_ERROR...open binary file
CERES_W_CLOSE_GAC_FAIL    WARNING...could not close GAC file
```