# *Appendix C*

# *Programmer's Guide*

*Appendix C - Programmer's Guide*

*This appendix provides instructions on how a user, who is proficient in the use of the Fortran computer program language, can modify existing DSSMATH compute functions and create new user defined functions. The DSSMATH program has been structured to accommodate the relatively easy addition of new functions. The user modifies one standard DSSMATH subroutine "USRFUN", in which a user defined function is identified and then takes one of the existing function subroutines and modifies it to incorporate the new function. The most difficult task is the understanding of the large number of variables that DSSMATH uses to store all the information dealing with the function parameters and data values. All the variables are documented in the common blocks in which they reside.*

*The following instructions are given as an example in which a simple user defined function is created. The name of the new function will be ABS(TX). The function will take the absolute value of all values in the TX variable.*

*Step 1. - Modify DSSMATH subroutine F.USRFUN*

*The shaded sections of code reflect the additions or modifications to the existing code to add the new user function ABS. Existing code is commented out; if it needs to be modified, it is duplicated and changes are made to the duplicated code.*

```
      SUBROUTINE USRFUN (ISTAT)
C
C     PROVIDES A LINK WITH FUNCTIONS DEVELOPED LOCALLY
C     ****************************************************************
C
C     SUBROUTINE AUTHOR:  DENNIS HUFF
C     HYDROLOGIC ENGINEERING CENTER
C     DEVELOPMENT DATE:   5 FEB 88
C     REVISION LOG:
C     SUBROUTINE PARAMETERS:
C
C     ISTAT    -  (ISTAT = -1) ===>  SOMETHING WENT WRONG WHILE
C                    PROCESSING THE COMMAND LINE FOR THE FILE NAME.
C               (ISTAT = 0 ) ===>  EVERYTHING IS OK.
$ADD C.IOCOM
$ADD C.CFUNCT
C
C     DETERMINE WHICH FUNCTION IS BEING USED
C
C     ****************************************************************

C     IF(CFUNEX(1:4).EQ.'USER') THEN
C        CALL USRSUB(ISTAT)
      IF(CFUNEX(1:4).EQ.'ABS') THEN
        CALL ABSFUN(ISTAT)

C
C     ****************************************************************
      ELSE
        ISTAT = -1
        WRITE(IFOUT,*)' ERROR - FUNCTION  ',CFUNEX(1:6),'  NOT VALID'
      ENDIF
C
```

```
      RETURN
      END
```

*Step 2. - Modify DSSMATH subroutine F.ACC*

*The second step is to pick one of the existing DSSMATH functions and modify it to create the new function ABS. In the example below, a copy of the existing file F.ACC is modified to produce the new F.ABSFUN subroutine. F.ACC was chosen because it is a function that is similar to function ABS. The following is a listing of the DSSMATH function ACC. The common blocks have been turned on so the variable definitions are visible.*

```
~~C      SUBROUTINE ACC(ISTAT)~~
~~C~~
~~C     COMPUTE A CUMULATIVE TIME-SERIES~~
~~C~~
~~C       TY = ACC(TX)~~
      SUBROUTINE ABSFUN(ISTAT)
C
C     COMPUTE ABSOLUTE OF TIME-SERIES VALUES
C
C       TY = ABS(TX)
C
C     ROUTINE CALLED FROM:
C
C     1.  SUBROUTINE FUNSUB
C
C     ******************************************************************
C
~~C     SUBROUTINE AUTHOR:  DENNIS HUFF~~
C     SUBROUTINE AUTHOR:  Alfredo E. Montalvo
C     HYDROLOGIC ENGINEERING CENTER
C     DAVIS CALIFORNIA
~~C     DEVELOPMENT DATE:   5  MAY 1987~~
C     DEVELOPMENT DATE:  21  JUNE 1990

C     REVISION LOG:
C
C
C
C     SUBROUTINE PARAMETERS:
C
C     -----OUTPUT-----
C
C     ISTAT    -  (ISTAT = -1) ===>  SOMETHING WENT WRONG WHILE
C                    PROCESSING
C              (ISTAT = 0 ) ===>  EVERYTHING IS OK.
C
C     LOCAL VARIABLES:
C
C     NFPR     -  NUMBER OF PARAMETERS IN THE FUNCTION
C     ILGPTY   -  ARRAY CONTAINING THE LEGITIMATE PARAMETER TYPES
C     NITSA1   -  NUMBER OF PAIRS OF PARAMETERS TO CHECK FOR STARTING
C              TIMES OF THE TIMES SERIES VARIABLES
C     ITSA1    -  INTEGER ARRAY USED TO POINT TO WHICH PARAMETERS
C              ARE TO BE CHECKED AGAINST
C     NITSA2   -  NUMBER OF PAIRS OF PARAMETERS TO CHECK FOR DATA
C              TIMES OF THE TIMES SERIES VARIABLES OCCURING AT
```

```
C              THE SAME TIMES
C     ITSA2    -  INTEGER ARRAY USED TO POINT TO WHICH PARAMETERS
C              ARE TO BE CHECKED AGAINST
C
C
```

```
C$ADD C.DSSDA1                                                    $
C                                                                 $
C     ****************************************************************     $
C     THIS COMMON BLOCK CONTAINS VARIABLES DEALING WITH THE DSS DATA AND      $
C     VARIABLE NAMES BEING USED TO HOLD THIS DATA                          $
C                                                                 $
      PARAMETER (NTS=10,NPF=5,NSCA=15,NPFC=30,NTSV=5000,NPFV=5000,          $
     & NPAT=NTS+NPF,NVAR=NTS+NPF+NSCA)                                     $
C                                                                 $
      COMMON /DSSDA1/ NHEADW(NPAT),HEADW(100,NPAT),NTYPE(NPAT),             $
     & NPNP(6,NPAT),NPATH(NPAT),ISTIME,ISDATE,IETIME,IEDATE,IHORIZ(NPAT)     $
     & ,TSDATE(NTSV,NTS),TSY(NTSV,NTS),PFX(NPFV,NPF),PFY(NPFV,NPF),          $
     & NCURVE(NPF),NDATA(NPAT),IPATH,SCALAR(NSCA),IQFLAG(NTSV,NTS)           $
C                                                                 $
      INTEGER HEADW                                                $
C                                                                 $
      COMMON /RDSSDA/ RBUFF                                          $
      REAL RBUFF(NTSV*2+100)                                         $
C                                                                 $
      COMMON /CDSSD1/ VARLBL,CPNP,CPATP,CSTIME,CSDATE,CETIME,CEDATE,          $
     & CUNITS,CTYPE,PATHNM,CARYLB                                     $
      CHARACTER VARLBL(NVAR)*6,CPNP(6,NPAT)*32,CPATP(6)*32,                 $
     & CSTIME*4,CSDATE*7,CETIME*4,CEDATE*7,                              $
     & CUNITS(2,NPAT)*8,CTYPE(2,NPAT)*8,PATHNM(NPAT)*80,                   $
     & CARYLB(NPFC,NPF)*8                                            $
C                                                                 $
      COMMON /LDSSDA/ LTWSET                                         $
      LOGICAL LTWSET                                               $
C                                                                 $
C     DESCRIPTION OF VARIABLES                                       $
C                                                                 $
C     NPAT      -  MAXIMUM NUMBER OF PATHNAMES                          $
C     NTS       -  MAXIMUM NUMBER OF TIME SERIES VECTORS                $
C     NPF       -  MAXIMUM NUMBER OF PAIRED FUNCTIONS                   $
C     NSCA      -  MAXIMUM NUMBER OF SCALARS                            $
C     NPFC      -  MAXIMUM NUMBER OF DEPENDENT VECTORS PER PAIRED       $
C                FUNCTION                                         $
C     NTSV      -  MAXIMUM SIZE OF TIME SERIES VECTOR                   $
C     NPFV      -  MAXIMUM SIZE OF PAIRED FUNCTION VECTOR               $
C     NVAR      -  MAXIMUM NUMBER OF USER VARIABLES                     $
C     NHEADW()   -  LENGTH OF HEADER ARRAY                              $
C     HEADW()    -  DSS DATA HEADER ARRAY                               $
C     VARLBL()  -  USER VARIABLE LABELS                                 $
C                FIRST NTS SLOTS POINT TO TS DATA                       $
C                NEXT NPF SLOTS POINT TO PF DATA                        $
C                LAST NSCA SLOTS POINT TO SCALARS                       $
C     NTYPE()   -  TYPE OF DATA THAT IS HELD BY PATHNAME                  $
C                1 - TIME SERIES REGULAR TIME INTERVAL                  $
C                2 - TIME SERIES IRREGULAR TIME INTERVAL                $
C                3 - PAIRED FUNCTION DATA                             $
C                4 - SCALAR VARIABLE                                $
C                5 - CHARACTER LABEL                                $
C     IHORIZ    -  VARIABLE NUMBER TO APPEAR ON THE HORIZONTAL AXIS FOR     $
C                PLOTTING (1 OR 2) - PAIRED DATA ONLY                   $
C     IQFLAG()  -  INTEGER ARRAY USED TO INDICATE THE FOLLOWING FOR       $
C                DATA VALUES STORED IN TSY() ARRAY.                     $
C                0 - NO QUALITY STATUS IMPLIED -                      $
C                1 - ESTIMATED VALUE                                $
C                2 - QUESTIONED VALUE                               $
C                3 - MISSING (UNDEFINED) VALUE FLAG                     $
C     ISTIME    -  TIME WINDOW FOR STARTING TIME  (MINUTES)             $
C     IETIME    -  TIME WINDOW FOR ENDING TIME    (MINUTES)             $
C     ISDATE    -  TIME WINDOW FOR STARTING DATE  (DAY COUNT)           $
C     IEDATE    -  TIME WINDOW FOR ENDING DATE    (DAY COUNT)           $
C     CSTIME    -  CHARACTER STARTING TIME (24 HOUR TIME)               $
```

```
C     CETIME    -  CHARACTER ENDING TIME   (24 HOUR TIME)                    $
C     CSDATE    -  CHARACTER STARTING DATE (MILITARY STYLE)                 $
C     CEDATE    -  CHARACTER ENDING DATE   (MILITARY STYLE)                 $
C     CPATP()   -  LAST PATHNAME ENTERED                                    $
C     IPATH     -  POINTER FROM CPATP TO CPNP                               $
C     CPNP()    -  PATHNAME PARTS A-F                                       $
C     NPNP()    -  LENGTH OF PATHNAME PARTS A-F                             $
C     CUNITS()  -  UNITS OF DSS DATA                                        $
C     CTYPE()   -  TYPE OF DSS DATA                                         $
C     PATHNM()  -  FULL PATHNAMES                                           $
C     NPATH()   -  LENGTH OF PATHNAMES                                      $
C     CARYLB()  -  LABELS FOR PAIRED FUNCTION DEPENDENT VECTORS             $
C     TSDATE()  -  TIME SERIES DATES (JULIAN DAYS WITH FRACTIONS)           $
C     TSY()     -  TIME SERIES DEPENDENT VARIABLE                           $
C     PFX()     -  PAIRED FUNCTION INDEPENDENT VARIABLE                     $
C     PFY()     -  PAIRED FUNCTION DEPENDENT VARIABLES                      $
C     NCURVE()  -  NUMBER OF DEPENDENT VARIABLES PER PAIRED FUNCTION        $
C     NDATA()   -  NUMBER OF ORDINATES PER CURVE (PF OR TS)                 $
C     LTWSET    -  LOGICAL FLAG TO INDICATE IF THE TIME WINDOW IS SET       $
C     RBUFF     -  DSS WORK SPACE                                           $
C     SCALAR()  -  SCALAR VARIABLES                                         $
C                                                                           $
C     ******************************************************************    $
C                                                                           $
      COMMON /TBUFFR/ XTBUFF(NTSV),ITBUFX(NTSV)                             $
C                                                                           $
C     XTBUFF() -  TEMPORARY BUFFER SPACE USED TO STORE A REAL VARIABLE      $
C     ITBUFX() -  TEMPORARY BUFFER SPACE USED TO STORE AN INTEGER VARIABLE  $
C
```

```
C$ADD C.IOCOM
C                                                                           $
C     ******************************************************************    $
C     THIS COMMON BLOCK HOLD VARIABLES DEALING WITH INPUT AND OUTPUT        $
C     UNIT NUMBERS AND OPERATING ENVIRONMENT                                $
C                                                                           $
      COMMON /IOCOM/ IFOUT,INPUT,IENVIR                                     $
C                                                                           $
C     VARIABLE DESCRIPTION                                                  $
C                                                                           $
C   INPUT   -  INPUT UNIT NUMBER                                            $
C   IFOUT   -  OUTPUT UNIT NUMBER                                           $
C   IENVIR  - =0 - INTERACTIVE EXECUTION                                    $
C           >0 - REAL TIME                                                  $
C           <0 - BATCH                                                      $
C     ******************************************************************    $
C                                                                           $
```

```
C$ADD C.CDEPND
C                                                                           $
C     ******************************************************************    $
C     THIS COMMON BLOCK CONTAINS VARIABLES DEALING WITH THE DEPENDENT       $
C     VARIABLE USED IN THE  COMPUTE COMMAND                                 $
C                                                                           $
      COMMON /CDEPND/ ICDPVA,ICDPTY,IDPVIF(5000)                            $
      COMMON /CCDEPN/ CDPVAR                                                $
      CHARACTER CDPVAR*6                                                    $
C                                                                           $
C     DESCRIPTION OF VARIABLES                                              $
C     CDPVAR   -  CHARACTER VARIABLE USED TO STORE DEPENDENT VARIABLE       $
C                 NAME                                                      $
C     ICDPVA   -  INTEGER VARIABLE USED TO SPECIFY TWO THINGS:              $
C                 0 - DEPENDENT VARIABLE CDPVAR HAS NOT BEEN DEFINED        $
```

```
16: C            0 < - INDEX OF CDPVAR LOCATION OF ITS DATA                    $
17: C     ICDPTY   -  INTEGER VARIABLE  USED TO SPECIFY THE TYPE OF DATA          $
18: C              STORED  IN DEPENDENT VARIABLE CDPVAR                          $
19: C     IDPVIF()  -  INTEGER ARRAY USED TO INDICATE IF THE DEPENDENT            $
20: C               VARIABLE SHOULD BE PROCESSED BASED ON THE IF FUNCTION        $
21: C               RESULTS                                                      $
22: C               0 - PROCESS THE DATA VALUE                                   $
23: C               1 - DO NOT PROCESS THE DATA VALUE                            $
24: C               2 - ONE OF THE IF FUNCTION PARAMETERS WAS MISSING            $
25: C                  AND THEREFORE DO NOT PROCESS THE DATA VALUE               $
26: C                                                                            $
27: C     ******************************************************************     $
28: C                                                                            $

 1: C$ADD C.CFUNCT
 2: C                                                                            $
 3: C     ******************************************************************     $
 4: C     THIS COMMON BLOCK CONTAINS VARIABLES DEALING WITH THE FUNCTIONS          $
 5: C     USED IN THE COMPUTE COMMAND                                              $
 6: C                                                                            $
 7:       COMMON /CFUNCT/ ICFUNP(10),ICFPTY(10),ICFPSG(10),XCFUNP(10),             $
 8:      . NFUNP,LFUNOP                                                           $
 9:       LOGICAL LFUNOP                                                          $
10:       COMMON /CCFUNC/ CFUNEX,CFUNP                                             $
11:       CHARACTER*10 CFUNEX,CFUNP(10)                                            $
12: C     DESCRIPTION OF VARIABLES                                                 $
13: C                                                                            $
14: C     CFUNEX   -  CHARACTER VARIABLE USED TO STORE THE FUNCTION NAME           $
15: C     CFUNP    -  CHARACTER VARIABLE ARRAY THAT HOLDS THE NAMES OF             $
16: C              FUNCTION PARAMETERS                                            $
17: C     ICFUNP   -  INTEGER ARRAY USED FOR TWO PURPOSES                          $
18: C               0 - INDICATES THAT FUNCTION PARAMETER IS A CONTANT            $
19: C            0 < - INDEX OF FUNCTION PARAMETER LOCATION                       $
20: C     ICFPTY   -  INTEGER ARRAY USED TO SPECIFY THE TYPE OF DATA STORED        $
21: C               SEE DESCRIPTION OF NTYPE() VARIABLE IN DSSDA1 COMMON          $
22: C     ICFPSG   -  INTEGER ARRAY USED TO HOLD SIGN OF PARAMETER VARIABLE        $
23: C               1 FOR POSITIVE AND -1 FOR NEGATIVE PARAMETER                  $
24: C     LFUNOP   -  LOGICAL VARIABLE USED TO INDICATE IF THE  FUNCTION           $
25: C              OPERATION IS BEING USED                                        $
26: C     NFUNP    -  NUMBER OF FUNCTION PARAMETERS                                $
27: C     XCFUNP   -  ARRAY USED FOR REAL NUMBER CONSTANT VALUE OF FUNCTION        $
28: C     ******************************************************************     $
29: C                                                                            $



48: C
49: C     SET SUBROUTINE PERCHK PARAMETER CHECKS FOR THIS FUNCTION
50:       DIMENSION ILGPTY(1),ITSA1(1),ITSA2(1)
51:       DATA NFPR,NITSA1,NITSA2 /1,0,0/
52:       DATA ILGPTY /12/
53: C
54:       CALL PERCHK( NFPR,ILGPTY,NITSA1,ITSA1,NITSA2,ITSA2,
55:      . ISTAT)
56:       IF(ISTAT.LT.0) GO TO 9000
57: C
58:       CALL STHDPV(1,ISTAT)
59:       IF(ISTAT.NE.0) GO TO 9000

60: C      IF(CTYPE(1,ICFUNP(1)).EQ.'PER-CUM') THEN
61: C      .   CTYPE(1,ICDPVA) = 'INST-CUM'
62: C      ELSE
63: C      .  CTYPE(1,ICDPVA) = CTYPE(1,ICFUNP(1))
64: C      ENDIF
```

```
60:      CTYPE(1,ICDPVA) = CTYPE(1,ICFUNP(1))

65:      CUNITS(1,ICDPVA)= CUNITS(1,ICFUNP(1))
66: C
67: C     PROCESS THE DATA

68: C      SUM = 0.
69:      DO 1000 I=1,NDATA(ICFUNP(1))
70: C        CHECK THE IF FUNCTION FLAG
71:      .  IF(IDPVIF(I).NE.0) THEN

72: C      .  .  TSY(I,ICDPVA)=SUM
72:            TSY(I,ICDPVA) = TSY(I,ICFUNP(1))

73:      .  .  IQFLAG(I,ICDPVA) = 0
74: C        CHECK TO SEE IF THE DATA IS MISSING
75:      .  ELSE IF(IQFLAG(I,ICFUNP(1)).GT.2) THEN
76:      .  .  IQFLAG(I,ICDPVA)=3
77:      .  .  TSY(I,ICDPVA)=-901
78: C        COMPUTE THE ACCUMULATION
79:      .  ELSE

80: C      .  .  SUM = SUM + TSY(I,ICFUNP(1))
81: C      .  .  TSY(I,ICDPVA)=SUM
81:            TSY(I,ICDPVA) = ABS(TSY(I,ICFUNP(1)))

82:      .  .  IQFLAG(I,ICDPVA)=0
83:      .  ENDIF
84:  1000 .  CONTINUE
85: C
86:      NDATA(ICDPVA)=NDATA(ICFUNP(1))
87: C
88:      NTYPE(ICDPVA) = NTYPE(ICFUNP(1))
89:      NHEADW(ICDPVA) = NHEADW(ICFUNP(1))
90: C
91:  9000 RETURN
92:      END
93: C
```