DECUS
PROGRAM LIBRARY

DECUS NO.	8-364
TITLE	EXTENDED MEMORY PATCH TO THE 3-WORD FLOATING POINT ARITHMETIC INTERPRETER
AUTHOR	Herbert Poppe
COMPANY	Lamont Geological Observatory of Columbia University Palisades, New York
DATE	November 1970
SOURCE LANGUAGE	PAL-D

NAME: Extended Memory Patch to the 3-Word Floating Point Arithmetic Interpreter

VERSION: 1

MOD: 0

DATE: November, 1970

PROGRAMMER: Herbert Poppe

ADDRESS: Lamont Geological Observatory of Columbia University
Palisades, New York 10964

PURPOSE:

To allow the 3-Word Floating Point Arithmetic Interpreter (DEC-08-YQYB) to reside in any memory field and to be entered from that field or any other memory field.

INTRODUCTION:

Few things are more frustrating to a computer user than to find that the systems software for his machine is tailored to the minimum configuration. His expensive expansion to extended memory and perhiperal devices are largely unsupported within the available system software. The patch described here enables the user to make efficient use of the floating-point arithmetic interpreter in an extended memory configuration. The patch and floating-point package (FPP) reside wholly within any memory field. The patch either modifies locations or occupies unused locations within the FPP. It may be used with any of the versions of the FPP (i.e., with or without the output controller, with or without the extended functions). The patch contains coding necessary for picking up the interpretive instructions from across field boundaries, for picking up and storing the contents of the effective address across field boundaries, and for returning control from the FPP across field boundaries to the calling field. The technique used to achieve these features, unlike DECUS No. 8-188 (Extended Memory Patch for Four Word Floating Point Package), does not require that certain locations common to the FPP be reserved in every field.

RESTRICTIONS:

This patch will not work with the previous version of the FPP, DEC-08-YQYA (DIGITAL 8-5-S). The only restriction on the use of the patched FPP is that the effective address of an interpretive instruction must be in the same field as the interpretive instruction. For example, if the patched

FPP resides in Field 3 and the list of interpretive instructions is in Field 1, then ADDR in FGET ADDR or DATA in FMPY I ADATA, where ADATA, DATA, is understood to be in Field 1.

CORE REQUIREMENTS:

The patch extensively modifies locations 56ØØ - 5774 and modifies or occupies locations 616Ø - 6177; 6271; 6336; 6373; 6375-6377; 6545-6563; 6655; and 716Ø - 7176.

OTHER PROGRAMS REQUIRED:

DEC-08-YQYB; 3-Word Floating Point Arithmetic Interpreter.

LOADING AND PATCHING:

Load the desired version of the FPP into the field in which it is to reside with the Binary Loader or the DISK/DECTAPE SYSTEM Binary Loader. Then load the patch into the same field. With the console switches, set the contents of location 5655 to 62N1, where N = memory field in which the FPP resides. This location is initially set to 6231 (Memory Field 3).

Note that the patch resets all the addresses in the interpretive subroutine table (from locations 6547 - 6563) to point to the dummy subroutine at EXIT 6 (loc. 6167). The user must set up these locations to point to the extended functions or his own interpretive routines. When patching is complete, the patched FPP may be saved on the systems device if available.

OPERATION:

On entry, the patched FPP uses the contents of the Data Field (DF) register to identify the calling field. Therefore, if necessary, the DF register must be set to the calling field at some time prior to entering the FPP. The following example illustrates this point and the method of entering the FPP. It is not necessary to clear the contents of the accumulator or link on entering the FPP and they will be cleared when control returns to the calling field.

```
          X0=30
          FENTR=JMS I 7
          FEXT=0000
          FGET=5000
          FPUT=6000
          FADD=1000

          FIELD 1
          *0007
          5600

          *0200
START,    CLA       CLL       /THIS CODE IN FIELD 1
          CDF+20
          TAD I     ADATA2    /PICK UP 3-WORD FLOATING
          DCA       DATA1     /POINT NUMBER FROM FIELD 2
          ISZ       ADATA2
          TAD I     ADATA2
          DCA       DATA1+1
          ISZ       ADATA2
          TAD I     ADATA2
          DCA       DATA1+2
          CDF+10              /SET DF TO CALLING FIELD
          CIF+X0              /FPP IN FIELD X
          FENTR
          FGET      DATA1
          FADD      DATA1
          FPUT      DATA1
          FEXT
          HLT                 /RETURN HERE WITH IF&DF=1

ADATA2,   DATA2
DATA1,    0000
          0000
          0000

          FIELD 2
          *0400
DATA2,    0002                /2.0
          2000
          0000

          $
```

```
                /EXTENDED MEMORY PATCH TO THE 3-WORD
                /FLOATING POINT ARITHMETIC INTERPRETER DEC-08-YQYB
                /(DOES NOT WORK WITH OLDER VERSION DEC-08-YQYA (DIGITAL 8-5$
                /HERBERT POPPE
                /LAMONT GEOLOGICAL OBSERVATORY
                /PALISADES, NEW YORK
                /OCTOBER 1970

                /FLOATING POINT PACKAGE MAY RESIDE IN ANY FIELD
                /FLOATING POINT PACKAGE MAY BE CALLED FROM ANY FIELD
                /SET X0 TO FIELD IN WHICH FFP IS TO RESIDE IF COMPILING
               /PATCH SOURCE TAPE; IE, X0=0030 FOR MEMORY FIELD 3
                /SET CONTENT OF LOCATION 5655 = 62N1, WHERE N = MEMORY
                /FIELD IN WHICH FPP IS TO RESIDE IF PATCHING WITH BINARY TAP
                /AFTER LOADING BINARY TAPE OF FPP INTO DESIRED FIELD,
                /LOAD PATCH OVER IT. THEN CHANGE LOCATION 5655 IF NECESSARY.

                /TO ENTER FLOATING POINT INTERPRETER (EXTENDED MEMORY VERSI
                /THE DATA FIELD (DF) REGISTER MUST BE SET TO THE CALLING
                /FIELD
                /       CALLING SEQUENCE EXAMPLE:
                /                       /THIS CODING IN FIELD 1
                /       .
                /       .
                /       CDF+10
                /       .
                /       .
                /       CIF+X0          /X0= FIELD FPP RESIDES IN
                /       FENTR           /FENTR=JMS I 0007
                /       .               /FP PSEUDO-INSTRUCTIONS
                /       .
                /       FEXT            /FEXT=0000
                /       .               /RETURN HERE WITH DF & IF = 1

                        EXP=0044;       HORD=0045;      LORD=0046
                       X0=0030;        MINUS2=6400;    OVER1=0043
                        OVER2=0047;     EX1=0040;       AC1H=0041
                        AC1L=0042;      ALIGN=6020;     FLDV=6264
                        DUNORM=6533;    DUBLAD=7053;    DNORM=6600

                        *5600
5600  0000  FPNT,    0000
5601  7600  MASK5,   7600            /GROUP 2 CLA
5602  1272          TAD     DATAF   /A CDF INSTR.
5603  6214          RDF             /GET CALLING FIELD
5604  3353         DCA     CFIELD
5605  1353          TAD     CFIELD
5606  3216          DCA     FARG
5607  1353          TAD     CFIELD
5610  3322          DCA     FLPT
5611  2353          ISZ     CFIELD
5612  2353          ISZ     CFIELD  /MAKE IT A CDF CIF
5613  7300  FPNT1,  CLA     CLL
5614  3043          DCA     OVER1
```

```
5615  3047            DCA     OVER2
5616  0000  FARG,     0000            /FIELD OF NEXT PSEUDO INSTR.
5617  1600            TAD I   FPNT
5620  3267            DCA     JUMP
5621  1267            TAD     JUMP
5622  0300            AND     PAGENO  /GET PAGE BIT
5623  7650            SNA     CLA     /PAGE ZERO?
5624  5227            JMP     .+3     /YES
5625  1201            TAD     MASK5   /NO
5626  0200            AND     FPNT    /C(FPNT)0-4 CONTAINS PAGE BITS
5627  3274            DCA     ADDR
5630  1277            TAD     MASK7   /GET 7 BIT ADDRESS
5631  0267            AND     JUMP
5632  1274            TAD     ADDR
5633  3274            DCA     ADDR
5634  1301            TAD     INDRCT  /INDIRECT BIT = 1?
5635  0267            AND     JUMP
5636  7650            SNA     CLA
5637  5242            JMP     LOOP01  /N0-GO ON
5640  1674            TAD I   ADDR    /YES, DEFER
5641  3274            DCA     ADDR
5642  2200  LOOP01,   ISZ     FPNT
5643  1674            TAD I   ADDR
5644  3040            DCA     EX1     /EXPONENT
5645  1274            TAD     ADDR
5646  3275            DCA     SAVE
5647  2275            ISZ     SAVE
5650  1675            TAD I   SAVE
5651  3041            DCA     AC1H    /HIGH ORDER MANTISSA
5652  2275            ISZ     SAVE
5653  1675            TAD I   SAVE
5654  3042            DCA     AC1L
5655  6231            CDF+X0          /X0 INDICATE FIELD FPP RESIDES IN
5656  1267            TAD     JUMP
5657  7106            CLL     RTL
5660  7006            RTL
5661  0276            AND     MASK3   /GET BITS 0-2; IE OPCODE
5662  1302            TAD     TABLE   /LOOKUP IN TABLE
5663  3270            DCA     JUMP2
5664  1670            TAD I   JUMP2
5665  3270            DCA     JUMP2
5666  5670            JMP I   JUMP2   /GO THERE
5667  0000  JUMP,     0000
5670  0000  JUMP2,    0000
5671  0000  GO0,      0000
5672  6201  DATAF,    CDF
5673  0000  GO2,      0000
5674  0000  ADDR,     0000
5675  0000  SAVE,     0000
5676  0017  MASK3,    0017
5677  0177  MASK7,    0177
5700  0200  PAGENO,   0200
5701  0400  INDRCT,   0400
5702  5703  TABLE,    .+1
5703  5747            EXIT            /TABLE USED IN INTERPRETING
```

```
5704  5735          FLAD              /BITS 0-2 OF PSEUDO
5705  5734          FLSU              /INSTRUCTION
5706  6171          FLMY
5707  6264          FLDV              /IF OPCODE=0, GO TO EXIT
5710  5713          FLGT              /AND INTERPRET BITS 8-11
5711  5722          FLPT
5712  5740          NORF
5713  1040  FLGT,   TAD     EX1       /FGET=5
5714  3044          DCA     EXP
5715  1041          TAD     AC1H
5716  3045          DCA     HORD
5717  1042          TAD     AC1L
5720  3046          DCA     LORD
5721  5213          JMP     FPNT1
5722  0000  FLPT,   0000              /CDF TO CALLING FIELD
5723  1044          TAD     EXP
5724  3674          DCA I   ADDR
5725  2274          ISZ     ADDR
5726  1045          TAD     HORD
5727  3674          DCA I   ADDR
5730  2274          ISZ     ADDR
5731  1046          TAD     LORD
5732  3674          DCA I   ADDR
5733  5213          JMP     FPNT1
5734  4742  FLSU,   JMS I   OPMINS    /FSUB=2; NEGATE OPERAND & ADD
5735  4743  FLAD,   JMS I   ALGN      /FLAD=1; FIRST ALIGN EXPONENTS
5736  4744          JMS I   UNORM     /LARGER OF THE TWO IS IN FAC
5737  4745          JMS I   DUBADD    /TRIPLE PREC. ADD SINCE BITS ARE SH
5740  4746  NORF,   JMS I   NORM      /NORMALIZE THE RESULT
5741  5213          JMP     FPNT1
5742  6400  OPMINS, MINUS2
5743  6020  ALGN,   ALIGN
5744  6533  UNORM,  DUNORM
5745  7053  DUBADD, DUBLAD
5746  6600  NORM,   DNORM
5747  1267  EXIT,   TAD     JUMP      /OPCODE=0
5750  0276          AND     MASK3     /ARE BITS 8-11=0?
5751  7440          SZA
5752  5356          JMP     .+4
5753  0000  CFIELD, 0000              /CDF CIF TO CALLING FIELD
5754  7300          CLA     CLL
5755  5600          JMP I   FPNT      /YES, EXIT
5756  1374          TAD     ACON6     /LOOKUP IN TABLE
5757  3270          DCA     JUMP2
5760  1670          TAD I   JUMP2
5761  3270          DCA     JUMP2
5762  1216          TAD     FARG      /SAVE FIELD OF NEXT PSEUDO-INSTR.
5763  3271          DCA     GO0
5764  1200          TAD     FPNT
5765  3273          DCA     GO2       /SAVE RETURN ADDRESS
5766  4670          JMS I   JUMP2     /CALL AS A SUBROUTINE
5767  7300          CLA     CLL       /IN CASE NOT CLEARED BY SUBR.
5770  1273          TAD     GO2
5771  3200          DCA     FPNT      /RESTORE RETURN ADDRESS
5772  1271          TAD     GO0
5773  5204          JMP     FPNT+4    /RESTORE MEMORY INSTR.
5774  6544  ACON6,  TABLE6-1          /CALLING CAN BE TO A DEPTH OF ONE
```

```
                        FPAC1=0052;     DMULT=6200;     NDRFLO=6513
                         FPUT=6000;      FMPY=3000;      FEXT=0000

                        *6160
6160  5613  RETN2,     FPNT1
6161  0000  SQUARE,    0000
6162  4407              JMS I    0007
6163  6052              FPUT     FPAC1
6164  3052              FMPY     FPAC1
6165  0000              FEXT
6166  5761              JMP I    SQUARE
6167  0000  EXIT6,     0000                 /DUMMY SUBROUTINE
6170  5767              JMP I    EXIT6
6171  7300  FLMY,      CLA      CLL         /FMPY=3
6172  4777              JMS I    FLONDR      /TEST EXPONENT UNDERFLOW
6173  3044              DCA      EXP
6174  4776              JMS I    MULT
6175  5760              JMP I    RETN2
6176  6200  MULT,      DMULT
6177  6513  FLONDR,    NDRFLO

                        *6271
6271  5375              JMP      ERROR       /DIVISION BY ZERO

                        *6336
6336  5740              NORF

                        *6373
6373  5613  RETN1,     FPNT1

                        FLAG=0061

                        *6375
6375  2061  ERROR,     ISZ      FLAG        /DIVISION BY ZERO--INCREMENT ERROR B
 6376  5773              JMP I    RETN1       /FAC UNCHANGED BY DIVISION BY ZERO
6377  5773              JMP I    RETN1

                        SQROOT=6673;    PSINF=EXIT6;    PCOSF=EXIT6
                        PATANF=EXIT6;   PEXPF=EXIT6;    PLOGF=EXIT6

                        *6545
6545  6161  TABLE6,    SQUARE              /TABLE FOR INTERPRETATION
6546  6673              SQROOT              /OF BITS 8-11
6547  6167              PSINF               /CONTAINS ABSOLUTE ADDRESSES
6550  6167              PCOSF               /OF PROGRAMS CALLED AS
6551  6167              PATANF              /SUBROUTINES
6552  6167              PEXPF               /EXIT6=A DUMMY OR NOP
6553  6167              PLOGF               /IF EXT. FCTS ARE NOT PRESENT
6554  6167              EXIT6               /PSINF,PCOSF,PATANF,PEXPF,AND
6555  6167              EXIT6               /PLOGF ARE SET=EXIT6
6556  6167              EXIT6
6557  6167              EXIT6
6560  6167              EXIT6
6561  6167              EXIT6
6562  6167              EXIT6
```

```
6563  6167            EXIT6

                      *6655
6655  7160  TESTHI,   BIGNEG

                      SHFTRT=6062;     AMOUNT=6151

                      *7160
              /TEST FOR LARGEST NEGATIVE NUMBER (MANTISSA=4000 000)
              /CALLED BY NORMALIZATION ROUTINE-DNORM-IF FOUND
              /SHIFT FLOATING AC RIGHT 1 BIT AND INCREMENT EXP
7160  0000    BIGNEG,  0000
7161  1046             TAD      LORD     /LOAD LOW ORDER WORD
7162  7640            SZA      CLA      /IS IT ZERO
7163  5760             JMP I    BIGNEG   /NO-CAN'T BE LARGEST NEG # RETURN
7164  1045             TAD      HORD     /YES-LOAD HI ORDER WORD TO CHECK FO
7165  7510             SPA               /SKIP IF # IS POSITIVE
7166  7041             CIA               /IF NEFATIVE, NEGATE IT
7167  7700             SMA      CLA      /IS # STILL NEGATIVE?
7170  5760             JMP I    BIGNEG   /NO--RETURN
7171  7360             CLA CMA CLL CML   /YES--PREPARE TO SHIFT FAC RIGHT 1 B
7172  3776             DCA I    AMONTC   /STORE -1 IN SHIFT COUNT
7173  4775             JMS I    BAKSFT   /JMS TO SHIFT FAC AND INCREMENT EXP
7174  5760             JMP I    BIGNEG   /RETURN
7175  6062    BAKSFT,  SHFTRT
7176  6151    AMONTC,  AMOUNT
```

```
ACON6   5774
AC1H    0041
AC1L    0042
ADDR    5674
ALGN    5743
ALIGN   6020
AMONTC  7176
AMOUNT  6151
BAKSFT  7175
BIGNEG  7160
CFIELD  5753
DATAF   5672
DMULT   6200
DNORM   6600
DUBADD  5745
DUBLAD  7053
DUNORM  6533
ERROR   6375
EXIT    5747
EXIT6   6167
EXP     0044
EX1     0040
FARG    5616
FEXT    0000
FLAD    5735
FLAG    0061
FLDV    6264
FLGT    5713
FLMY    6171
FLONDR  6177
FLPT    5722
FLSU    5734
FMPY    3000
FPAC1   0052
FPNT    5600
FPNT1   5613
FPUT    6000
GO0     5671
GO2     5673
HORD    0045
INDRCT  5701
JUMP    5667
JUMP2   5670
LOOP01  5642
LORD    0046
MASK3   5676
MASK5   5601
MASK7   5677
MINUS2  6400
MULT    6176
NDRFLO  6513
NORF    5740
NORM    5746
OPMINS  5742
```

```
OVER1   0043
OVER2   0047
PAGENO  5700
PATANF  6167
PCOSF   6167
PEXPF   6167
PLOGF   6167
PSINF   6167
RETN1   6373
RETN2   6160
SAVE    5675
SHFTRT  6062
SQROOT  6673
SQUARE  6161
TABLE   5702
TABLE6  6545
TESTHI  6655
UNORM   5744
X0      0030
```