```
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRR           TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
```

```
MM       MM  TTTTTTTTTT  HH       HH  DDDDDDDD    EEEEEEEEEE  XX      XX  PPPPPPPP
MM       MM  TTTTTTTTTT  HH       HH  DDDDDDDD    EEEEEEEEEE  XX      XX  PPPPPPPP
MMMM   MMMM      TT      HH       HH  DD      DD  EE          XX      XX  PP      PP
MMMM   MMMM      TT      HH       HH  DD      DD  EE          XX      XX  PP      PP
MM  MM  MM      TT      HH       HH  DD      DD  EE            XX  XX    PP      PP
MM  MM  MM      TT      HH       HH  DD      DD  EE            XX  XX    PP      PP
MM      MM      TT      HHHHHHHHHHH  DD      DD  EEEEEEEE      XX      PPPPPPPP
MM      MM      TT      HHHHHHHHHHH  DD      DD  EEEEEEEE      XX      PPPPPPPP
MM      MM      TT      HH       HH  DD      DD  EE            XX  XX    PP
MM      MM      TT      HH       HH  DD      DD  EE            XX  XX    PP
MM      MM      TT      HH       HH  DD      DD  EE          XX      XX  PP          ....
MM      MM      TT      HH       HH  DD      DD  EE          XX      XX  PP          ....
MM      MM      TT      HH       HH  DDDDDDDD    EEEEEEEEEE  XX      XX  PP          ....
MM      MM      TT      HH       HH  DDDDDDDD    EEEEEEEEEE  XX      XX  PP          ....

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

(2) 51 HISTORY ; Detailed Current Edit History
(3) 75 DECLARATIONS ; Declarative Part of Module
(4) 240 MTH$DEXP - Standard Double Precision Floating EXP
(5) 294 MTH$DEXP_R6 - Special DEXP routine

```
0000     1          .TITLE  MTH$DEXP        ; Double Precision Floating Exponential
0000     2                                  ; Function (DEXP)
0000     3          .IDENT /1-014/          ; File: MTHDEXP.MAR  Edit: RNH1014
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28 ;
0000    29 ; FACILITY: MATH LIBRARY
0000    30 ;++
0000    31 ; ABSTRACT:
0000    32 ;
0000    33 ;       MTH$DEXP is a function which returns the double floating point
0000    34 ; exponential of its double precision floating point argument.
0000    35 ; The call is standard call-by-reference.
0000    36 ;
0000    37 ;--
0000    38 ;
0000    39 ; VERSION: 1
0000    40 ;
0000    41 ; HISTORY:
0000    42 ; AUTHOR:
0000    43 ;       Peter Yuo, 15-Oct-76: Version 0
0000    44 ;       Mary Payne and Jud Leonard, 23-JAN-78: Version 1
0000    45 ;
0000    46 ; MODIFIED BY:
0000    47 ;
0000    48 ;
0000    49 ;
```

```
0000     51          .SBTTL  HISTORY ; Detailed Current Edit History
0000     52
0000     53 ; Edit History for Version 1 of MTH$DEXP
0000     54 ;
0000     55 ; 1-1    Split single and double precision routines into two parts;
0000     56 ;        Used more accurate and faster algorithms provided by M. Payne.
0000     57 ;        JMT 23-Jan-78
0000     58 ; 1-3    Fixed bug causing unexpected integer overflow.  JMT 24-Mar-78
0000     59 ; 1-4    Change from rational approx. to Chebyshev approx. for
0000     60 ;        more accuracy with overhang.  MHP 18-May-78
0000     61 ; 1-5    Detect underflow properly.  TNH 16-June-78
0000     62 ; 1-6    Remove extra instruction.  TNH 16-June-78
0000     63 ; 1-7    Move .ENTRY mask to module header.  TNH 14-Aug-78
0000     64 ; 1-008 - Update version number and copyright notice.  JBS 16-NOV-78
0000     65 ; 1-009 - Change symbols MTH__FLOUNDMAT and MTH__FLOOVEMAT to
0000     66 ;          MTH$K_FLOUNDMAT and MTH$K_FLOOVEMAT.  JBS 07-DEC-78
0000     67 ; 1-010 - Add "_" to the PSECT directive.  JBS 22-DEC-78
0000     68 ; 1-011 - Declare externals.  SBL 17-May-1979
0000     69 ; 1-012 - Use only through R6.  SBL 21-Sept-1979
0000     70 ; 1-013 - Included logic for large arguments (where EMOD loses
0000     71 ;          significance).  RNH 23-JUN-81
0000     72 ; 1-014 - Changed W^ to G^ on calls th MTH$$SIGNAL and MTH$$JACKET_TST
0000     73 ;          RNH 09-Sept-1981
```

```
                          0000     75          .SBTTL  DECLARATIONS            ; Declarative Part of Module
                          0000     76
                          0000     77 ;
                          0000     78 ; INCLUDE FILES:        MTHJACKET.MAR
                          0000     79 ;
                          0000     80
                          0000     81 ;
                          0000     82 ; EXTERNAL SYMBOLS:
                          0000     83 ;
                          0000     84          .DSABL  GBL
                          0000     85          .EXTRN  MTH$K_FLOUNDMAT
                          0000     86          .EXTRN  MTH$K_FLOOVEMAT
                          0000     87          .EXTRN  MTH$$SIGNAL
                          0000     88          .EXTRN  MTH$$JACKET_TST
                          0000     89
                          0000     90 ;
                          0000     91 ; EQUATED SYMBOLS:
                          0000     92
                 0000407C 0000     93          ACMASK  = ^M<IV, R2, R3, R4, R5, R6>    ; .ENTRY mask + int ovf enable
                 000000BB 0000     94          X_273   =       ^0273           ; Extension for operand in EMODD
                 00004370 0000     95          SD_60   =       ^F60.
                 00004240 0000     96          SD_12   =       ^F12.
                 000043F0 0000     97          SD_120  =       ^F120.
                          0000     98 ;
                          0000     99 ; MACROS:
                          0000    100          $SFDEF                          ; define SF$ (stack frame) symbols
                          0000    101 ;
                          0000    102 ; PSECT DECLARATIONS:
                          0000    103
                 00000000 0000    104          .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                          0000    105                                          ; program section for math routines
                          0000    106 ;
                          0000    107 ; OWN STORAGE:  none
                          0000    108 ;
                          0000    109 ; CONSTANTS:
                          0000    110
                          0000    111 ;
                          0000    112 ; Table to be used for scaling. These constants here have been
                          0000    113 ; verified by M. Payne using her program ROOT16 on PDP-10.
                          0000    114 ;
 0000 4080                0000    115 TABHI:  .WORD   ^0040200,0              ; 2**(0/16) = 1.0
 0000 0000                0004    116         .WORD   0,0
 AAC3 4085                0008    117         .WORD   ^0040205,^0125303       ; 2**(1/16)
 487B 67CC                000C    118         .WORD   ^0063714,^0044173
 95C1 408B                0010    119         .WORD   ^0040213,^0112701       ; 2**(2/16)
 8BD7 E3EA                0014    120         .WORD   ^0161752,^0105727
 C3D3 4091                0018    121         .WORD   ^0040221,^0141723       ; 2**(3/16)
 11C3 73AB                001C    122         .WORD   ^0071653,^0010703
 37F0 4098                0020    123         .WORD   ^0040230,^0033760       ; 2**(4/16)
 B8A9 518D                0024    124         .WORD   ^0050615,^0134251
 F532 409E                0028    125         .WORD   ^0040236,^0172462       ; 2**(5/16)
 A112 6091                002C    126         .WORD   ^0060221,^0120422
 FED6 40A5                0030    127         .WORD   ^0040245,^0177326       ; 2**(6/16)
 5139 A9B1                0034    128         .WORD   ^0124661,^0050471
 583E 40AD                0038    129         .WORD   ^0040255,^0054076       ; 2**(7/16)
 A14B EA42                003C    130         .WORD   ^0165102,^0120513
 04F3 40B5                0040    131         .WORD   ^0040265,^0002363       ; 2**(8/16)
```

```
DE65 33F9  0044   132          .WORD   ^0031771,^0157145
08A3 40BD  0048   133          .WORD   ^0040275,^0004243       ; 2**(9/16)
0C37 9F58  004C   134          .WORD   ^0117530,^0006067
672A 40C5  0050   135          .WORD   ^0040305,^0063452       ; 2**(10/16)
06DB 1155  0054   136          .WORD   ^0010525,^0003333
248C 40CE  0058   137          .WORD   ^0040316,^0022214       ; 2**(11/16)
8481 151F  005C   138          .WORD   ^0012437,^0102201
44FC 40D7  0060   139          .WORD   ^0040327,^0042374       ; 2**(12/16)
9D6B CAD6  0064   140          .WORD   ^0145326,^0116553
CCDE 40E0  0068   141          .WORD   ^0040340,^0146336       ; 2**(13/16)
94E1 EC2A  006C   142          .WORD   ^0166052,^0112341
C0C6 40EA  0070   143          .WORD   ^0040352,^0140306       ; 2**(14/16)
2439 E7DD  0074   144          .WORD   ^0163735,^0022071
257D 40F5  0078   145          .WORD   ^0040365,^0022575       ; 2**(15/16)
86CC 1524  007C   146          .WORD   ^0012444,^0103314
           0080   147
           0080   148
0000 0000  0080   149 TABLO:   .WORD   0,0                     ; DECIMAL: 0.D0
0000 0000  0084   150          .WORD   0,0
2E48 2326  0088   151          .WORD   ^0021446,^0027110       ; DECIMAL: 0.2252169616881804D-17
0000 0000  008C   152          .WORD   0,0
20D0 A348  0090   153          .WORD   ^0121510,^0020320       ; DECIMAL:-0.2712242510500122D-17
0000 0000  0094   154          .WORD   0,0
3F62 23D8  0098   155          .WORD   ^0021730,^0037542       ; DECIMAL: 0.5861402647731367D-17
0000 0000  009C   156          .WORD   0,0
8D5A 245E  00A0   157          .WORD   ^0022136,^0106532       ; DECIMAL: 0.1206457647223494D-16
0000 0000  00A4   158          .WORD   0,0
BEE1 A424  00A8   159          .WORD   ^0122044,^0137341       ; DECIMAL:-0.8930877995013540D-17
0000 0000  00AC   160          .WORD   0,0
1A10 A32F  00B0   161          .WORD   ^0121457,^0015020       ; DECIMAL:-0.2373071989573779D-17
0000 0000  00B4   162          .WORD   0,0
D9F8 A3E6  00B8   163          .WORD   ^0121746,^0154770       ; DECIMAL:-0.6257240830881880D-17
0000 0000  00BC   164          .WORD   0,0
4D06 A477  00C0   165          .WORD   ^0122167,^0046406       ; DECIMAL:-0.1340620676392399D-16
0000 0000  00C4   166          .WORD   0,0
AF00 A402  00C8   167          .WORD   ^0122002,^0127400       ; DECIMAL:-0.7084371812598154D-17
0000 0000  00CC   168          .WORD   0,0
0756 A38B  00D0   169          .WORD   ^0121613,^0003526       ; DECIMAL:-0.3768379065187162D-17
0000 0000  00D4   170          .WORD   0,0
EE5A A360  00D8   171          .WORD   ^0121540,^0167132       ; DECIMAL:-0.3048384309613603D-17
0000 0000  00DC   172          .WORD   0,0
6580 A2BC  00E0   173          .WORD   ^0121274,^0062600       ; DECIMAL:-0.1276624235300040D-17
0000 0000  00E4   174          .WORD   0,0
32C0 2308  00E8   175          .WORD   ^0021410,^0031300       ; DECIMAL: 0.1845830375854930D-17
0000 0000  00EC   176          .WORD   0,0
40B4 23BB  00F0   177          .WORD   ^0021673,^0040264       ; DECIMAL: 0.5075495866202897D-17
0000 0000  00F4   178          .WORD   0,0
EE77 23B1  00F8   179          .WORD   ^0021661,^0167167       ; DECIMAL: 0.4822843060675619D-17
0000 0000  00FC   180          .WORD   0,0
           0100   181
           0100   182 ;
           0100   183 ; Constants used in evaluation of polynomials - small arguments
           0100   184 ;
12C8 38D0  0100   185 DXPTB1:  .WORD   ^0034320,^0011310
CA18 5B74  0104   186          .WORD   ^0055564,^0145030       ; DECIMAL:   0.2480427857745020D-04
1381 3A50  0108   187          .WORD   ^0035120,^0011601
962F 4F31  010C   188          .WORD   ^0047461,^0113057       ; DECIMAL:   0.1984369200268758D-03
```

```
                    0110   189          .WORD   ^0035666,^0005540
                    0114   190          .WORD   ^0120720,^0161470       ; DECIMAL:   0.1388888879690042D-02
                    0118   191          .WORD   ^0036410,^0104210
                    011C   192          .WORD   ^0072406,^0171775       ; DECIMAL:   0.8333333262370290D-02
                    0120   193          .WORD   ^0037052,^0125252
                    0124   194          .WORD   ^0125252,^0162167       ; DECIMAL:   0.4166666666679500D-01
                    0128   195          .WORD   ^0037452,^0125252
                    012C   196          .WORD   ^0125253,^0000535       ; DECIMAL:   0.1666666666667437D-00
                    0130   197          .WORD   ^0040000,0
                    0134   198          .WORD   0,0                     ; DECIMAL:   0.5000000000000000D-00
                    0138   199          .WORD   ^0040177,^0177777
                    013C   200          .WORD   ^0177777,^0177777       ; DECIMAL:   0.1000000000000000D+01
                    0140   201          .WORD   ^0040200,0
                    0144   202          .WORD   0,0                     ; DECIMAL:   0.1000000000000000D+01
 00000009           0148   203 DX^LN1=<.-DXPTB1>/8                      ; no. of entries in table
                    0148   204
                    0148   205 ;
                    0148   206 ; Constants used in evaluation of polynomial - regular args
                    0148   207 ;
                    0148   208
                    0148   209
                    0148   210 DXPTAB: .WORD   ^0023261,^0061157
                    014C   211          .WORD   ^0021767,^0125016       ; DECIMAL:   0.3077130709430240D-15
                    0150   212          .WORD   ^0025177,^0164725
                    0154   213          .WORD   ^0143237,^0103423       ; DECIMAL:   0.5682419384166091D-13
                    0158   214          .WORD   ^0027041,^0102211
                    015C   215          .WORD   ^0074021,^0127075       ; DECIMAL:   0.9181219559808114D-11
                    0160   216          .WORD   ^0030656,^0141777
                    0164   217          .WORD   ^0033217,^0147273       ; DECIMAL:   0.1271587192556359D-08
                    0168   218          .WORD   ^0032435,^0112533
                    016C   219          .WORD   ^0076722,^0074645       ; DECIMAL:   0.1467610032291993D-06
                    0170   220          .WORD   ^0034143,^0054106
                    0174   221          .WORD   ^0134045,^0011314       ; DECIMAL:   0.1355080777949815D-04
                    0178   222          .WORD   ^0035565,^0176757
                    017C   223          .WORD   ^0176026,^0026166       ; DECIMAL:   0.9383847928089872D-03
                    0180   224          .WORD   ^0037061,^0071027
                    0184   225          .WORD   ^0173721,^0147572       ; DECIMAL:   0.4332169878499658D-01
                    0188   226          .WORD   0,0
                    018C   227          .WORD   0,0                     ; DECIMAL:   0
 00000009           0190   228 DXPLN=<.-DXPTAB>/8                       ; no. of entries in table
                    0190   229
                    0190   230
                    0190   231
                    0190   232 D_16LOG2_E:                             ; LOG2(E) * 16
            AA3B 42B8   0190   233          .WORD   ^0041270,^0125073
            17F0 295C   0194   234          .WORD   ^0024534,^0013760
                    0198   235 D_LN2_OV_16_HI:
C800F7D1 72173E31   0198   236          .QUAD   ^XC800F7D172173E31      ; Hi 45 bits of ln2/16
                    01A0   237 D_LN2_OV_16_LO:
73003C76 357927EF   01A0   238          .QUAD   ^X73003C76357927EF      ; Low bits of ln2/16
```

```
                     01A8    240          .SBTTL  MTH$DEXP - Standard Double Precision Floating EXP
                     01A8    241
                     01A8    242
                     01A8    243 ;++
                     01A8    244 ; FUNCTIONAL DESCRIPTION:
                     01A8    245 ;
                     01A8    246 ; EXP  - Double precision floating point function
                     01A8    247 ;
                     01A8    248 ; Uses a Chebyshev approximation, with overhang on last step.
                     01A8    249 ;
                     01A8    250 ;
                     01A8    251 ; CALLING SEQUENCE:
                     01A8    252 ;
                     01A8    253 ;       Exponential.wd.v = MTH$EXP(x.rd.r)
                     01A8    254 ;
                     01A8    255 ; INPUT PARAMETERS:
                     01A8    256
           00000004  01A8    257          LONG = 4                            ; define longword multiplier
           00000004  01A8    258          x = 1 * LONG                        ; Contents of x is the argument
                     01A8    259
                     01A8    260 ; IMPLICIT INPUTS:      none
                     01A8    261 ;
                     01A8    262 ; OUTPUT PARAMETERS:
                     01A8    263 ;
                     01A8    264 ;       VALUE:  floating exponential of the argument
                     01A8    265 ;
                     01A8    266 ; IMPLICIT OUTPUTS:     none
                     01A8    267 ;
                     01A8    268 ; SIDE EFFECTS:
                     01A8    269 ;
                     01A8    270 ; Signals: MTH$_FLOOVEMAT if X > 88.028 with reserved operand in R0/R1 (copied
                     01A8    271 ; to the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).  Associated
                     01A8    272 ; message is: "FLOATING OVERFLOW IN MATH LIBRARY". Result is reserved operand
                     01A8    273 ; -0.0 unless a user supplied (or any) error handler changes CHF$L_MCH_R0/R1.
                     01A8    274 ;       MTH$_FLOUNDMAT if X =< -89.416 and caller has hardware enable set.
                     01A8    275 ; The result is set to +0.0. Associated message is: "FLOATING UNDERFLOW IN MATH
                     01A8    276 ; LIBRARY"
                     01A8    277 ;
                     01A8    278 ; NOTE: This procedure disables floating point underflow, enable integer
                     01A8    279 ; overflow, causes no floating overflow or other arithmetic traps, and
                     01A8    280 ; preserves enables across the call.
                     01A8    281 ;
                     01A8    282 ;--
                     01A8    283
                     01A8    284
                 407C 01A8    285          .ENTRY  MTH$DEXP, ACMASK            ; standard call-by-reference entry
                     01AA    286                                              ; disable DV (and FU), enable IV
                     01AA    287          MTH$FLAG_JACKET                     ; flag that this is a jacket procedure
                     01AA
 6D  00000000'GF  9E 01AA              MOVAB   G^MTH$$JACKET_HND, (FP)
                     01B1                                                     ; set handler address to jacket
                     01B1                                                     ; handler
                     01B1
                     01B1    288                                              ; in case of an error in special JSB
                     01B1    289                                              ; routine
  50  04 BC      70 01B1    290          MOVD    @x(AP), R0                  ; R0/R1 = user's arg
        01      10 01B5    291          BSBB    MTH$DEXP_R6                 ; R0/R1 = special EXP(R0/R1)
```

```
                                H 4
04  01B7   292          RET                              ; return - result in R0/R1
```

```
                                01B8   294          .SBTTL  MTH$DEXP_R6 - Special DEXP routine
                                01B8   295
                                01B8   296 ; Special DEXP - used by the standard, and direct interfaces.
                                01B8   297 ;
                                01B8   298 ; CALLING SEQUENCE:
                                01B8   299 ;       save anything needed in R0:R6
                                01B8   300 ;       MOVD    ...,R0                  ; input in R0
                                01B8   301 ;       JSB     MTH$DEXP_R6
                                01B8   302 ;       return with result in R0/R1
                                01B8   303
                                01B8   304 ; Note: This routine is written to avoid causing any integer overflows,
                                01B8   305 ; floating overflows, or floating underflows or divide by 0 conditions,
                                01B8   306 ; whether enabled or not.
                                01B8   307 ;
                                01B8   308 ; REGISTERS USED:
                                01B8   309 ;       R0/R1 - floating argument, then result
                                01B8   310 ;       R2/R3 - temp
                                01B8   311 ;       R5 - integer scratch
                                01B8   312 ;       R6 - integer part of X * LG2(E) * 16   (16I+J)
                                01B8   313
                                01B8   314
                                01B8   315 MTH$DEXP_R6::                           ; special DEXP routine
                                01B8   316 MTH$DEXP_R7::                           ; Release 1 name
    52    50    8000 8F    AB   01B8   317          BICW3   #^X8000, R0, R2         ; Preliminary test for over/underflow
                                01BE   318                                         ; R2 = 1st word of !X!
    53    52    3E00 8F    A3   01BF   319          SUBW3   #^X3E00, R2, R3         ; R3 = 4 + unbiased exponent
          53    05B0 8F    B1   01C4   320          CMPW    #^X5B0, R3              ; Compare !X! with 88
                      60    1F   01C9   321          BLSSU   SMTST                   ; to more tests if LSSU
                                01CB   322                                         ;  else, -4 < unbiased exp < 8
                                01CB   323                                         ;  no exceptions in EMODD or APPROX
                                01CB   324
                4280 8F    52   B1   01CB   325          CMPW    R2, #^X4280             ; Check for loss of significance in
                                01D0   326                                         ; EMOD ( !X! >= 2**4
                      20    19   01D0   327          BLSS    EVAL                    ; No loss of significance
                                01D2   328
                                01D2   329 ;
                                01D2   330 ; !X! >= 2**4.  EMOD will lose significance so the interger and fractional
                                01D2   331 ; parts of X*16/ln2 must be obtained in seperate steps.
                                01D2   332 ;
          52    50      BB AF   65   01D2   333          MULD3   D_16LOG2_E, R0, R2      ; Get integer part of X*16/ln2 in
                56    52   6A   01D7   334          CVTDL   R2, R6                  ;    R6 (=I+J) as a longword and in
                52    56   6E   01DA   335          CVTLD   R6, R2                  ;    R2/R3 in D format
          54    52      B8 AF   65   01DD   336          MULD3   D_LN2_OV_16_HI, R2, R4  ; Get fraction part of X*16/ln2 =
                50    54   62   01E2   337          SUBD    R4, R0                  ;    16/ln2*[ X - (I+J)*ln2/16 ]
                52      B8 AF   64   01E5   338          MULD    D_LN2_OV_16_LO, R2      ;    in R0/R1.
                50    52   62   01E9   339          SUBD    R2, R0                  ;
                50      A1 AF   64   01EC   340          MULD    D_16LOG2_E, R0          ;
                              08   11   01F0   341          BRB     APPROX          ;
                                01F2   342
50    56    50    BB 8F    9B AF   74   01F2   343 EVAL:   EMODD   D_16LOG2_E, #X_273, R0, R6, R0
                                01FA   344                                         ; get X*16*LG2(E) with
                                01FA   345                                         ;  integer part in R6 (=16I+J)
                                01FA   346                                         ;  fraction in R0/R1
                                01FA   347
          FF48 CF    08    50   75   01FA   348 APPROX: POLYD   R0,#DXPLN-1,DXPTAB      ; use Chebyshev series
                                0200   349                                         ;  with last coefficient 0
                                0200   350                                         ;  so that last ADDD has overhang
```

```
                                0200   351
        55   56   FFFFFFF0 8F   CB   0200   352          BICL3   #-16, R6, R5                ; R5 = J
             50     FDF3 CF45   64   0208   353          MULD    TABHI[R5], R0               ; else MUL by 2**(J/16)
             50     FE6D CF45   60   020E   354          ADDD    TABLO[R5], R0               ; add in LO of 2**(J/16)
             50     FDE7 CF45   60   0214   355          ADDD    TABHI[R5], R0               ; and then HI of 2**(J/16)
                                0021A  356
                       56   0F   CA   021A   357          BICL    #15, R6                     ; R6 = I
                            0B   13   021D   358          BEQL    20$                         ; if I=0, then done
                       50   6046 7E   021F   359          MOVAQ   (R0)[R6], R0                ; shift I to EXP position
                                0223   360                                                    ; MUL by 2**I by exponent addition
                    007F 8F    50   B1   0223   361          CMPW    R0, #^X7F               ; test for over/underflow
                               3D   15   0228   362          BLEQ    EXCEPT                  ; see what exception is if neg or = 0
                               05   022A   363 20$:     RSB                                 ; otherwise return result in R0
                                022B   364
                                022B   365 SMTST:
                               12   19   022B   366          BLSS    20$                     ; exception if exp+4 > 11
                    2400 8F    52   B1   022D   367          CMPW    R2, #^X2400             ; eliminate underflow from APPROX1
                               07   19   0232   368          BLSS    10$                     ; bypass if E**ARG = 1
                                0234   369
                                0234   370 ;+
                                0234   371 ; Use Chebyshev series for small arg
                                0234   372 ;-
                                0234   373
             FEC6 CF    08    50   75   0234   374          POLYD   R0,#DXPLN1-1,DXPTB1     ;Use Chebyshev series
                                023A   375                                                    ;  last term is 1; this will
                                023A   376                                                    ;  give desired overhang.
                               05   023A   377          RSB                                 ; answer is OK, return
                                023B   378
                       50   08   70   023B   379 10$:     MOVD    S^#1, R0                ; E**X is 1, store it
                               05   023E   380          RSB                                 ; and return
                                023F   381
                                023F   382
                                023F   383 ;
                                023F   384 ; Handlers for software detected over/underflow conditions follow
                                023F   385 ;
                       50   73   023F   386 20$:     TSTD    R0                          ; if big ARG > 0 goto OVERFLOW
                               28   18   0241   387          BGEQ    OVER
                                0243   388 ;
                                0243   389 ; Underflow; if user has FU set, signal error.  Always return 0.0
                                0243   390 ;
                                0243   391 UNDER:
                       52   DC   0243   392          MOVPSL  R2                          ; R2 = user's or jacket routine's PSL
        00000000'GF    00   FB   0245   393          CALLS   #0, G^MTH$$JACKET_TST       ; R0 = TRUE if JSB from jacket routine
                  04   50   E9   024C   394          BLBC    R0, 10$                     ; branch if user did JSB
                52   04   AD   3C   024F   395          MOVZWL  SF$W_SAVE_PSW(FP), R2       ; get user PSL saved by CALL
                       50   7C   0253   396 10$:     CLRQ    R0                          ; R0 = result. LIB$SIGNAL will save in
                                0255   397                                                ; CHF$L_MCH_R0/R1 so any handler can fixup
                0D 52   06   E1   0255   398          BBC     #6, R2, 20$                 ; has user enabled floating underflow?
                       6E   DD   0259   399          PUSHL   (SP)                        ; yes, return PC from special routine
                7E   00'8F   9A   025B   400          MOVZBL  #MTH$K_FLOUNDMAT, -(SP)     ; trap code for hardware floating underflow
                                025F   401                                                ; convert to MTH$_FLOUNDMAT (32-bit VAX-11
                                025F   402                                                ; exception code)
        00000000'GF    02   FB   025F   403          CALLS   #2, G^MTH$$SIGNAL           ; signal (condition, PC)
                               05   0266   404 20$:     RSB                                 ; return
                                0267   405
                                0267   406 EXCEPT:
                       56   D5   0267   407          TSTL    R6                          ; test sign of I; if I < 0
```

```
                     D8   19  0269   408          BLSS    UNDER                       ; go to underflow handler
                              026B   409 ;
                              026B   410 ; Signal floating overflow, return reserved operand, -0.0
                              026B   411 ;
                              026B   412 OVER:                                        ; else process for overflow
                     6E   DD  026B   413          PUSHL   (SP)                        ; return PC from special routine
                   7E   00'8F 9A  026D   414          MOVZBL  #MTH$K_FLOOVEMAT, -(SP) ; hardware floating overflow
                50   01   0F  79  0271   415          ASHQ    #15, #T, R0             ; R0/R1 = result = reserved operand
                              0275   416                                              ; -0.0. R0/0 will be copied to
                              0275   417                                              ; signal mechanism vector (CHF$L_MCH_R0/R1)
                              0275   418                                              ; so can be fixed up by any error
                              0275   419                                              ; handler
         00000000'GF    02   FB  0275   420          CALLS   #2, G^MTH$$SIGNAL       ; signal (condition, PC)
                              05  027C   421          RSB                             ; return - R0 restoredd from CHF$L_MCH_R0/R1
                              027D   422
                              027D   423
                              027D   424          .END
```

```
ACMASK          =  0000407C
APPROX             000001FA R      02
DXPLN           =  00000009
DXPLN1          =  00000009
DXPTAB             00000148 R      02
DXPTB1             00000100 R      02
D_16LOG2_E         00000190 R      02
D_LN2_OV_16_HI     00000198 R      02
D_LN2_OV_16_LO     000001A0 R      02
EVAL               000001F2 R      02
EXCEPT             00000267 R      02
LONG            =  00000004
MTH$$JACKET_HND    ********   X    02
MTH$$JACKET_TST    ********   X    00
MTH$$SIGNAL        ********   X    00
MTH$DEXP           000001A8 RG     02
MTH$DEXP_R6        000001B8 RG     02
MTH$DEXP_R7        000001B8 RG     02
MTH$K_FLOOVEMAT    ********   X    00
MTH$K_FLOUNDMAT    ********   X    00
OVER               0000026B R      02
SF$W_SAVE_PSW   =  00000004
SMTST              0000022B R      02
TABHI              00000000 R      02
TABLO              00000080 R      02
UNDER              00000243 R      02
X               =  00000004
X_273           =  000000BB
```

+----------------+
! Psect synopsis !
+----------------+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_MTH$CODE	0000027D (637.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

+-------------------------+
! Performance indicators !
+-------------------------+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.09	00:00:00.56
Command processing	135	00:00:00.70	00:00:03.99
Pass 1	133	00:00:02.03	00:00:07.53
Symbol table sort	0	00:00:00.04	00:00:00.05
Pass 2	87	00:00:01.02	00:00:03.84
Symbol table output	4	00:00:00.03	00:00:00.03
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	395	00:00:03.94	00:00:16.16

The working set limit was 1050 pages.
8608 bytes (17 pages) of virtual memory were used to buffer the intermediate code.

```
There were 10 pages of symbol table space allocated to hold 59 non-local and 5 local symbols.
484 source lines were read in Pass 1, producing 13 object records in Pass 2.
9 pages of virtual memory were used to define 8 macros.

                                        +----------------------------+
                                        ! Macro library statistics !
                                        +----------------------------+

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                   4

88 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:MTHDEXP/OBJ=OBJ$:MTHDEXP MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC$:
```

MTHDCOSH LIS

MTHDMIN LIS

MTHDLOG LIS

MTHDSINCO LIS

MTHDATANH LIS

MTHDNINT LIS

MTHDSQRT LIS

MTHDCONJG LIS

MTHDINT LIS

MTHDMAX1 LIS

MTHDSIGN LIS

MTHDIM LIS

MTHDMOD LIS

MTHDSINH LIS

MTHDEXP LIS

MTHDFLOOR LIS

MTHDPROD LIS