```
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    LL          OOOOOO      GGGGGGGG
MM      MM  TTTTTTTTTT  HH      HH  DDDDDDDD    LL          OOOOOO      GGGGGGGG
MMMM  MMMM      TT      HH      HH  DD      DD  LL        OO      OO  GG
MMMM  MMMM      TT      HH      HH  DD      DD  LL        OO      OO  GG
MM  MM  MM      TT      HH      HH  DD      DD  LL        OO      OO  GG
MM  MM  MM      TT      HH      HH  DD      DD  LL        OO      OO  GG
MM      MM      TT      HHHHHHHHHH  DD      DD  LL        OO      OO  GG
MM      MM      TT      HHHHHHHHHH  DD      DD  LL        OO      OO  GG
MM      MM      TT      HH      HH  DD      DD  LL        OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  DD      DD  LL        OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  DD      DD  LL        OO      OO  GG      GG      ....
MM      MM      TT      HH      HH  DD      DD  LL        OO      OO  GG      GG      ....
MM      MM      TT      HH      HH  DDDDDDDD    LLLLLLLLLL    OOOOOO      GGGGGG      ....
MM      MM      TT      HH      HH  DDDDDDDD    LLLLLLLLLL    OOOOOO      GGGGGG      ....

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

```
0000     1          .TITLE  MTH$DLOG        ; Floating Point Natural and Common
0000     2                                  ; Logarithm Functions (DLOG,DLOG10)
0000     3          .IDENT /2-003/          ; File: MTHDLOG.MAR  Edit: PDG2003
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$DLOG and MTH$DLOG10 are functions which return the double precision
0000    34 ; floating point natural or common logarithm of  their  double  precision
0000    35 ; floating point argument. The call is standard call-by-reference.
0000    36 ; MTH$DLOG_R8  and  MTH$DLOG10_R8 are special routines which are the same
0000    37 ; as MTH$DLOG and MTH$DLOG10 except a faster  non-standard  JSB  call  is
0000    38 ; used with the argument in R0 and no registers are saved.
0000    39 ;
0000    40 ;--
0000    41 ;
0000    42 ; VERSION: 01
0000    43 ;
0000    44 ; HISTORY:
0000    45 ; AUTHOR:
0000    46 ;       Peter Yuo, 15-Oct-76: Version 01
0000    47 ;
0000    48 ; MODIFIED BY:
0000    49 ;
0000    50 ; 01-1  Peter Yuo, 22-May-77
0000    51 ;
0000    52 ; VERSION: 02
0000    53 ;
0000    54 ; HISTORY:
0000    55 ; AUTHOR:
0000    56 ;       Bob Hanek, 18-Jun-81: Version 02
0000    57 ;
```

```
                     0000     58 ;
                     0000     59 ;
```

```
0000     61          .SBTTL  HISTORY ; Detailed Current Edit Histor
0000     62
0000     63
0000     64 ; ALGORITHMIC DIFFERENCE FROM FP-11C ROUTINE:
0000     65 ;       1. Uses POLYD so greater accuracy.
0000     66 ;
0000     67 ; Edit History for Version 01 of MTH$DLOGDLOG10
0000     68 ;
0000     69 ; 01-2  May 20, 1977    P. Yuo
0000     70 ;       Multiplication of EXPONENT(X) by ln(2) is done after POLY instead of
0000     71 ;       before, so one less register is used.
0000     72 ; 01-4  May 22, 1977    P. Yuo
0000     73 ;       Code saving after code review
0000     74 ; 01-6  MTH$$ERROR changed to MTH$$SIGNAL.
0000     75 ;       MTH$_... changed to MTH__...
0000     76 ;       Changed error handling mechanism. Put error result in R0:R1 before
0000     77 ;       calling MTH$$SIGNAL in order to allow user modify error result.
0000     78 ; 01-8  Add Rich Lary's code bums for speed...  JMT 26-Jan-78
0000     79 ; 01-9  Move .ENTRY mask definition to module header.  TNH 14-Aug-78
0000     80 ; 1-010 - Update version number and copyright notice.  JBS 16-NOV-78
0000     81 ; 1-011 - Change MTH__LOGZERGEG to MTH$K_LOGZERNEG.  JBS 07-DEC-78
0000     82 ; 1-012 - Add "_" to the PSECT directive.  JBS 22-DEC-78
0000     83 ; 1-013 - Add comment explaining code trickery.  SBL 14-Feb-1979
0000     84 ; 1-014 - Declare externals.  SBL 17-May-1979
0000     85 ;
0000     86 ;
0000     87 ; Edit History for Version 02 of MTH$DLOGDLOG10
0000     88 ;
0000     89
0000     90 ; 2-001 - Add MTH$DLOG2.  RNH 08-Aug-1981
0000     91 ; 2-002 - Correct entry logic for JSB entries.  Use G^ addressing for
0000     92 ;         externals.  SBL 24-Aug-1981
0000     93 ; 2-003 - Change D_FHI to the global symbol MTH$$AB_D_FHI.  PDG 3-Nov-81
```

```
                     0000     95          .SBTTL  DECLARATIONS    ; Declarative Part of Module
                     0000     96
                     0000     97 ;
                     0000     98 ; INCLUDE FILES:        MTHJACKET.MAR
                     0000     99 ;
                     0000    100
                     0000    101 ;
                     0000    102 ; EXTERNAL SYMBOLS:
                     0000    103 ;
                     0000    104          .DSABL  GBL
                     0000    105          .EXTRN  MTH$K_LOGZERNEG
                     0000    106          .EXTRN  MTH$$SIGNAL
                     0000    107          .EXTRN  MTH$$AB_ALOG
                     0000    108
                     0000    109 ;
                     0000    110 ; EQUATED SYMBOLS:
                     0000    111
000041FC             0000    112          ACMASK = ^M<IV, R2, R3, R4, R5, R6, R7, R8>
                     0000    113                                   ; register save mask and IV enable
00004080             0000    114          SD_1           =       ^F1.0   ; short floating literal 1.0
                     0000    115
                     0000    116 ;
                     0000    117 ; MACROS:       none
                     0000    118 ;
                     0000    119 ; PSECT DECLARATIONS:
                     0000    120
                00000000     121          .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                     0000    122                                   ; program section for math routines
                     0000    123 ;
                     0000    124 ; OWN STORAGE:  none
                     0000    125 ;
                     0000    126 ; CONSTANTS:
                     0000    127
                     0000    128 ;
                     0000    129 ;
                     0000    130 ;   The  D_FHI  table is accessed by an index obtained from the MTH$$AB_ALOG
                     0000    131 ;   table.  The MTH$$AB_ALOG table is located in MTHALOG.MAR.  Indices
                     0000    132 ;   between 0 and 13 inclusive are used to access entries 0 through 13
                     0000    133 ;   respectively.  For these indecies, the first three items of the
                     0000    134 ;   corresponding entry are FHI, LN_FHI_LO and LN_FHI_HI .  The last two
                     0000    135 ;   items for these entries are not used.  Indices between 14 and 27
                     0000    136 ;   inclusive access entries 13 through 0 respectively.  For these indecies,
                     0000    137 ;   the last three items in the corresponding entry are LN_FHI_HI, LN_FHI_LO
                     0000    138 ;   and FHI.  The first two items for these entries are not used.
                     0000    139 ;
                     0000    140
                     0000    141 MTH$$AB_D_FHI::
                     0000    142 ; Entry  0
00000000 4F9040ED    0000    143          .QUAD   ^X000000004F9040ED      ; .18539905548095703E+01
7C182E8E B8ED339C    0008    144          .QUAD   ^X7C182E8EB8ED339C      ; .18244885037187055E-07
0000789E 0A04401E    0010    145          .QUAD   ^X0000789E0A04401E      ; .61734035439258150E+00
81067183 B292339C    0018    146          .QUAD   ^X81067183B292339C      ; .18241995369217608E-07
00000000 149E400A    0020    147          .QUAD   ^X00000000149E400A      ; .53937709331512451E+00
                     0028    148 ; Entry  1
00000000 1D4340CF    0028    149          .QUAD   ^X000000001D4340CF      ; .16180804967880249E+01
3D2E5B7A D371B40F    0030    150          .QUAD   ^X3D2E5B7AD371B40F      ;-.33487088150728428E-07
60000B76 652B3FF6    0038    151          .QUAD   ^X60000B76652B3FF6      ; .48124060168191818E+00
```

```
0B9C4628 D935B40F  0040   152          .QUAD   ^X0B9C4628D935B40F      ;-.33492331660229068E-07
00000000 364F401E  0048   153          .QUAD   ^X000000000364F401E     ; .61801618337631226E+00
                   0050   154 ; Entry  2
00000000 68D440BA  0050   155          .QUAD   ^X0000000068D440BA      ; .14563241004943848E+01
A61156F1 0CD33306  0058   156          .QUAD   ^XA61156F10CD33306      ; .78027427538038965E-08
60 08AC0 77FF3FC0  0060   157          .QUAD   ^X60008AC077FF3FC0      ; .37591551369422405E+00
CDAD85CD A4CC3306  0068   158          .QUAD   ^XCDAD85CDA4CC3306      ; .78372974978005425E-08
00000000 C8F9402F  0070   159          .QUAD   ^X00000000C8F9402F      ; .68666034936904907E+00
                   0078   160 ; Entry  3
00000000 AD1D40AB  0078   161          .QUAD   ^X00000000AD1D40AB      ; .13412204980850220E+01
0E198034 C7A5B262  0080   162          .QUAD   ^X0E198034C7A5B262      ;-.33000814305226938E-08
4000E416 501E3F96  0088   163          .QUAD   ^X4000E416501E3F96      ; .29358002218498314E+00
4E4F98EB EC3AB262  0090   164          .QUAD   ^X4E4F98EBEC3AB262      ;-.33014787786590467E-08
00000000 DEF5403E  0098   165          .QUAD   ^X00000000DEF5403E      ; .74558955430984497E+00
                   00A0   166 ; Entry  4
00000000 1DA240A1  00A0   167          .QUAD   ^X000000001DA240A1      ; .12587168216705322E+01
CD6919A2 775532F6  00A8   168          .QUAD   ^XCD6919A2775532F6      ; .71731088180994794E-08
8000616B 9D723F6B  00B0   169          .QUAD   ^X8000616B9D723F6B      ; .23009279937366500E+00
C455AB15 49CA32F6  00B8   170          .QUAD   ^XC455AB1549CA32F6      ; .71679314343157716E-08
00000000 61B9404B  00C0   171          .QUAD   ^X000000061B9404B       ; .79445987939834595E+00
                   00C8   172 ; Entry  5
00000000 8BD24099  00C8   173          .QUAD   ^X000000008BD24099      ; .11995794773101807E+01
670D0B74 F961339A  00D0   174          .QUAD   ^X670D0B74F961339A      ; .18041364037815043E-07
4000E4BA 569D3F3A  00D8   175          .QUAD   ^X4000E4BA569D3F3A      ; .18197104176084622E+00
AFFF96B7 FE36339A  00E0   176          .QUAD   ^XAFFF96B7FE36339A      ; .18043562357555396E-07
00000000 687B4055  00E8   177          .QUAD   ^X00000000687B4055      ; .83362549543380737E+00
                   00F0   178 ; Entry  6
00000000 FFA64093  00F0   179          .QUAD   ^X00000000FFA64093      ; .11562392711639404E+01
A6B0F8DA A4AF33D2  00F8   180          .QUAD   ^XA6B0F8DAA4AF33D2      ; .24522108738420825E-07
A0006714 A8273F14  0100   181          .QUAD   ^XA0006714A8273F14      ; .14517270628599022E+00
B4C359CD AACF33D2  0108   182          .QUAD   ^XB4C359CDAACF33D2      ; .24524892962309998E-07
00000000 6850405D  0110   183          .QUAD   ^X000000006850405D      ; .86487293243408203E+00
                   0118   184 ; Entry  7
00000000 C18C408F  0118   185          .QUAD   ^X00000000C18C408F      ; .11230940818786621E+01
EE2F7A5B B6A5330E  0120   186          .QUAD   ^XEE2F7A5BB6A5330E      ; .83070168326007080E-08
A0009BED BF403EED  0128   187          .QUAD   ^XA0009BEDBF403EED      ; .11608744121413395E+00
20E25DDF AD3A330E  0130   188          .QUAD   ^X20E25DDFAD3A330E      ; .83048753355949145E-08
00000000 F1154063  0138   189          .QUAD   ^X00000000F1154063      ; .89039736986160278E+00
                   0140   190 ; Entry  8
00000000 5B39408C  0140   191          .QUAD   ^X000000005B39408C      ; .10965338945388794E+01
C08B3EA8 8FDBB3B1  0148   192          .QUAD   ^XC08B3EA88FDBB3B1      ;-.20670930291477513E-07
A0001EBC BB5A3EBC  0150   193          .QUAD   ^XA0001EBCBB5A3EBC      ; .92154220641148754E-01
D0AD3C6F 7941B3B1  0158   194          .QUAD   ^XD0AD3C6F7941B3B1      ;-.20660652275383675E-07
00000000 76814069  0160   195          .QUAD   ^X0000000076814069      ; .91196447610855103E+00
                   0168   196 ; Entry  9
00000000 B2B24089  0168   197          .QUAD   ^X00000000B2B24089      ; .10757658481597900E+01
5EC72E37 64A1340B  0170   198          .QUAD   ^X5EC72E3764A1340B      ; .32454981566347323E-07
80007615 92373E95  0178   199          .QUAD   ^X8000761592373E95      ; .73032792369019717E-01
7ACE3A06 5ACA340B  0180   200          .QUAD   ^X7ACE3A065ACA340B      ; .32446032444473518E-07
00000000 F853406D  0188   201          .QUAD   ^X00000000F853406D      ; .92957037687301636E+00
                   0190   202 ; Entry 10
00000000 B4DF4087  0190   203          .QUAD   ^X00000000B4DF4087      ; .10602072477340698E+01
26F6B5B8 ADAD33B9  0198   204          .QUAD   ^X26F6B5B8ADAD33B9      ; .21615814400350767E-07
60009CEF 78593E6F  01A0   205          .QUAD   ^X60009CEF78593E6F      ; .58464384127510982E-01
E061C397 B27D33B9  01A8   206          .QUAD   ^XE061C397B27D33B9      ; .21618002968246588E-07
00000000 76554071  01B0   207          .QUAD   ^X0000000076554071      ; .94321185350418091E+00
                   01B8   208 ; Entry 11
```

```
00000000 54244086  01B8   209          .QUAD   ^X0000000054244086      ; .10494427680969238E+01
E9F0E38D B48AB412  01C0   210          .QUAD   ^XE9F0E38DB48AB412      ;-.34157476452222763E-07
A0006AC5 AB9B3E45  01C8   211          .QUAD   ^XA0006AC5AB9B3E45      ; .48259360406838425E-01
437DC492 B075B412  01D0   212          .QUAD   ^X437DC492B075B412      ;-.34153763436452798E-07
00000000 F0604073  01D8   213          .QUAD   ^X00000000F0604073      ; .95288658142089844E+00
                   01E0   214 ; Entry 12
00000000 38494085  01E0   215          .QUAD   ^X0000000038494085      ; .10407801866531372E+01
AE5B4241 5CE33328  01E8   216          .QUAD   ^XAE5B42415CE33328      ; .98000725797821 71E-08
A000A9A3 B8363E23  01F0   217          .QUAD   ^XA000A9A3B8363E23      ; .39970601583469545E-01
D4DA866C 39CC3328  01F8   218          .QUAD   ^XD4DA866C39CC3328      ; .97920289877412402E-08
00000000 F8264075  0200   219          .QUAD   ^X00000000F8264075      ; .96081769466400146E+00
                   0208   220 ; Entry 13
00000000 6EE94084  0208   221          .QUAD   ^X000000006EE94084      ; .10346347093582153E+01
5D360D07 4E8B3345  0210   222          .QUAD   ^X5D360D074E8B3345      ; .11484767848640602E-07
6000D68B 76593E0B  0218   223          .QUAD   ^X6000D68B76593E0B      ; .34048415117204911E-01
281BA6FA 33443345  0220   224          .QUAD   ^X281BA6FA33443345      ; .11478566232851673E-07
00000000 6E2A4077  0228   225          .QUAD   ^X000000006E2A4077      ; .96652472019195557E+00
                   0230   226
                   0230   227 ;
                   0230   228 ;  Polynomial constants tables
                   0230   229 ;
                   0230   230
                   0230   231 LOGTAB1:                                 ; Constants for q(z).  Generated using
                   0230   232                                         ; eq. 6.3.10 of Hart et. al. (sin(2a)
                   0230   233                                         ; = 1/32)
1E51DE52 4D00BECD  0230   234          .QUAD   ^X1E51DE524D00BECD      ; C9 =  -0.10024452856511271
44C1F2BD 0E683EE4  0238   235          .QUAD   ^X44C1F2BD0E683EE4      ; C8 =   0.11135560980588577
9B9BEC78 FFDBBEFF  0240   236          .QUAD   ^X9B9BEC78FFDBBEFF      ; C7 =  -0.12499973121073342
4BB28A46 49143F12  0248   237          .QUAD   ^X4BB28A4649143F12      ; C6 =   0.14285690397225510
6C93AD01 AAAA8F2A  0250   238          .QUAD   ^X6C93AD01AAAA8F2A      ; C5 =  -0.16666666680280835
C92CCE8D CCCC3F4C  0258   239          .QUAD   ^XC92CCE8DCCCC3F4C      ; C4 =   0.20000000010208756
DCE9FFFF FFFF8F7F  0260   240          .QUAD   ^XDCE9FFFFFFFF8F7F      ; C3 =  -0.24999999999996883
A0A5AAAA AAAA3FAA  0268   241          .QUAD   ^XA0A5AAAAAAAA3FAA      ; C2 =   0.33333333333331553
00000000 0000C000  0270   242          .QUAD   ^X0000000000000C000     ; C1 =  -0.50000000000000000
00000000 00000000  0278   243          .QUAD   ^X0000000000000000      ; C0 =   0.00000000000000000
         0000000A  0280   244 LOGLEN1 = .-LOGTAB1/8                     ; no. of floating point entries
                   0280   245
                   0280   246
                   0280   247 LOGTAB2:                                 ; Constants for p(z*z).  Generated using
                   0280   248                                         ; eq. 6.3.11 of Hart et. al. (sin(2a) =
                   0280   249                                         ; (b - 1)/(b + 1) where b = 2**(1/7))
88B900ED 70B23F3B  0280   250          .QUAD   ^X88B900ED70B23F3B      ; C5 =  0.18304708605445 1500
5D2C3E00 8D3C3F63  0288   251          .QUAD   ^X5D2C3E008D3C3F63      ; C4 =  0.22221845749308 2486
185BC1CE 49243F92  0290   252          .QUAD   ^X185BC1CE49243F92      ; C3 =  0.28571429124626 5538
1CEFCCC4 CCCC3FCC  0298   253          .QUAD   ^X1CEFCCC4CCCC3FCC      ; C2 =  0.39999999996049620
AB02AAAA AAAA402A  02A0   254          .QUAD   ^XAB02AAAAAAAA402A      ; C1 =  0.66666666666667879
00000000 00004100  02A8   255          .QUAD   ^X0000000000004100      ; C0 =  2.00000000000000000
         00000006  02B0   256 LOGLEN2 = .-LOGTAB2/8
                   02B0   257
                   02B0   258
                   02B0   259
                   02B0   260 ;+  The "128" in the constants is used to shift the unbiased exponent
                   02B0   261 ;   right 7 places so that the rightmost bit is at bit 0.
                   02B0   262 ;-
                   02B0   263
                   02B0   264 D_LN_2_HI:
7200F7B1 72173CB1  02B0   265          .QUAD   ^X7200F7B172173CB1      ; (Hi 48 bits of ln2)/128
```

```
                     02B8   266 D_LN_2_LO:
8ECEAF27 75E62B81    02B8   267         .QUAD   ^X8ECEAF2775E62B81      ; (Low bits of ln2)/128
                     02C0   268 D_DLOG10_E:                              ; LOG10(e)
         5BD8 3FDE   02C0   269         .WORD   ^0037736,^0055730
         2872 A937   02C4   270         .WORD   ^0124467,^0024162
                     02C8   271 D_INV_LN2_CONS:                          ; convert natural log to log base 2
17F1295C AA3B40B8    02C8   272         .DOUBLE 1.44269504088896340735992468100189213742666
                     02D0   273
                     02D0   274
                     02D0   275
```

```
                     02D0    277          .SBTTL  MTH$DLOG  - Standard Double Precision Floating DLOG
                     02D0    278
                     02D0    279
                     02D0    280 ;++
                     02D0    281 ; FUNCTIONAL DESCRIPTION:
                     02D0    282 ;
                     02D0    283 ; DLOG  - single precision floating point function
                     02D0    284 ;
                     02D0    285 ; DLOG(X) is computed using the following approximation technique:
                     02D0    286 ;
                     02D0    287 ;       If X =< 0, error.  Otherwise
                     02D0    288 ;
                     02D0    289 ;       Let X = f * (2**n), where 1/2 <= f < 1
                     02D0    290 ;
                     02D0    291 ;       If n is greater than or equal to 1 than
                     02D0    292 ;             set N = n - 1 and F = 2*f.
                     02D0    293 ;          Else
                     02D0    294 ;             set N = n and F = f.
                     02D0    295 ;
                     02D0    296 ;       Then  ln(x) = N*ln2 + ln(F)
                     02D0    297 ;
                     02D0    298 ;       If |F - 1| < 2**-5 then
                     02D0    299 ;             ln(F) =  W + W*P(W), where W = F - 1 and P
                     02D0    300 ;             is a polynomial of degree 8.
                     02D0    301 ;          Else
                     02D0    302 ;             ln(F) = ln(FHI) + Z*Q(Z*Z), where FHI is ob-
                     02D0    303 ;             tained by table look-up, Q is a polynomial of
                     02D0    304 ;             degree 5 and Z = (F - FHI)/(F + FHI)
                     02D0    305 ;
                     02D0    306 ;       NOTE: The quantities ln(FHI) and ln2 are used in the above
                     02D0    307 ;             equations in two parts - a high part (containing the
                     02D0    308 ;             high order bits) and a low part (containing the low
                     02D0    309 ;             order bits.  In the code the high and low parts of the
                     02D0    310 ;             constants are indicated by a _HI and _LO suffix respec-
                     02D0    311 ;             tively.  The values were chosen such that N*LN_2_HI +
                     02D0    312 ;             LN_FHI_HI is exactly representable.
                     02D0    313 ;
                     02D0    314 ;
                     02D0    315 ; CALLING SEQUENCE:
                     02D0    316 ;
                     02D0    317 ;       logarithm.wd.v = MTH$DLOG(x.rd.r)
                     02D0    318 ;
                     02D0    319 ; INPUT PARAMETERS.
                     02D0    320
00000004             02D0    321          LONG = 4                        ; define longword multiplier
00000004             02D0    322          x = 1 * LONG                    ; Contents of x is the argument
                     02D0    323
                     02D0    324 ; IMPLICIT INPUTS:       none
                     02D0    325 ;
                     02D0    326 ; OUTPUT PARAMETERS:
                     02D0    327 ;
                     02D0    328 ;       VALUE:  double precision floating logarithm of the argument
                     02D0    329 ;
                     02D0    330 ; IMPLICIT OUTPUTS:      none
                     02D0    331 ;
                     02D0    332 ; COMPLETION CODES:      none
                     02D0    333 ;
```

```
                        02D0    334 ; SIDE EFFECTS:
                        02D0    335 ;
                        02D0    336 ; Signals: MTH$_LOGZERNEG if |X| =< 0.0 with reserved operand in R0/R1
                        02D0    337 ; (copied to the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                        02D0    338 ; Associated message is: "LOGARITHM OF ZERO OR NEGATIVE VALUE". Result is
                        02D0    339 ; reserved operand -0.0 unless a user supplied (or any) error handler changes
                        02D0    340 ; CHF$L_MCH_R0/R1.
                        02D0    341 ;
                        02D0    342 ; NOTE: This procedure disables floating point underflow, enables integer
                        02D0    343 ; overflow, causes no floating overflow or other arithmetic traps, and
                        02D0    344 ; preserves enables across the call.
                        02D0    345 ;
                        02D0    346 ;---
                        02D0    347
                        02D0    348
                   41FC 02D0    349         .ENTRY  MTH$DLOG, ACMASK            ; standard call-by-reference entry
                        02D2    350                                             ; disable DV (and FU), enable IV
                        02D2    351         MTH$FLAG_JACKET                     ; flag that this is a jacket procedure
                        02D2
      6D  00000000'GF    9E 02D2              MOVAB   G^MTH$$JACKET_HND, (FP)
                        02D9                                                    ; set handler address to jacket
                        02D9                                                    ; handler
                        02D9
                        02D9    352                                             ; in case of an error in special JSB
                        02D9    353                                             ; routine
        50    04 BC    70 02D9    354         MOVD    @x(AP), R0                  ; R0/R1 = arg
              34    10 02DD    355         BSBB    MTH$DLOG_R8                 ; call special DLOG rountine
                    04 02DF    356         RET                                 ; return - result in R0
                        02E0    357
```

```
                                02E0   359          .SBTTL  MTH$DLOG10 - Standard Single Precision Floating Common log
                                02E0   360
                                02E0   361 ;++
                                02E0   362 ; FUNCTIONAL DESCRIPTION:
                                02E0   363 ;
                                02E0   364 ; DLOG10  - single precision floating point function
                                02E0   365 ;
                                02E0   366 ; DLOG10(X) is computed as DLOG10(E) * DLOG(X).
                                02E0   367 ;
                                02E0   368 ; See description of MTH$DLOG
                                02E0   369 ;
                                02E0   370 ; CALLING SEQUENCE:
                                02E0   371 ;
                                02E0   372 ;       logarithm_base_10.wd.v = MTH$DLOG10(x.rd.r)
                                02E0   373 ;
                                02E0   374 ; INPUT PARAMETERS:
                                02E0   375
                    00000004    02E0   376          LONG = 4                        ; define longword multiplier
                    00000004    02E0   377          x = 1 * LONG                    ; Contents of x is the argument
                                02E0   378
                                02E0   379 ;
                                02E0   380 ; SIDE EFFECTS: See description of MTH$DLOG
                                02E0   381 ;
                                02E0   382 ;--
                                02E0   383
                                02E0   384
                        41FC    02E0   385          .ENTRY  MTH$DLOG10, ACMASK      ; standard call-by-reference entry
                                02E2   386                                          ; disable DV (and FU), enable IV
                                02E2   387          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                02E2
        6D  00000000'GF   9E    02E2                MOVAB   G^MTH$$JACKET_HND, (FP)
                                02E9                                                ; set handler address to jacket
                                02E9                                                ; handler
                                02E9
                                02E9   388                                          ; in case of an error in special JSB
                                02E9   389                                          ; routine
    50    04 BC       70        02E9   390          MOVD    @x(AP), R0              ; R0/R1 = arg
                15    10        02ED   391          BSBB    MTH$DLOG10_R8           ; call special DLOG10 routine
                      04        02EF   392          RET                             ; return - result in R0
                                02F0   393
```

```
                                02F0   395          .SBTTL  MTH$DLOG2  - Standard Single Precision Floating Common logarithm
                                02F0   396
                                02F0   397 ;++
                                02F0   398 ; FUNCTIONAL DESCRIPTION:
                                02F0   399 ;
                                02F0   400 ; DLOG2  - double precision floating point function
                                02F0   401 ;
                                02F0   402 ; DLOG2(X) is computed as DLOG2(E) * DLOG(X).
                                02F0   403 ;
                                02F0   404 ; See description of MTH$DLOG
                                02F0   405 ;
                                02F0   406 ; CALLING SEQUENCE:
                                02F0   407 ;
                                02F0   408 ;       logarithm_base_2.wd.v = MTH$DLOG2(x.rd.r)
                                02F0   409 ;
                                02F0   410 ; INPUT PARAMETERS:
                                02F0   411
                       00000004 02F0   412          LONG = 4                        ; define longword multiplier
                       00000004 02F0   413          x = 1 * LONG                    ; Contents of x is the argument
                                02F0   414
                                02F0   415 ;
                                02F0   416 ; SIDE EFFECTS: See description of MTH$DLOG
                                02F0   417 ;
                                02F0   418 ;--
                                02F0   419
                                02F0   420
                           41FC 02F0   421          .ENTRY  MTH$DLOG2, ACMASK       ; standard call-by-reference entry
                                02F2   422                                          ; disable DV (and FU), enable IV
                                02F2   423          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                02F2
      6D    00000000'GF    9E 02F2               MOVAB   G^MTH$$JACKET_HND, (FP)
                                02F9                                                ; set handler address to jacket
                                02F9                                                ; handler
                                02F9
                                02F9   424                                          ; in case of an error in special JSB
                                02F9   425                                          ; routine
      50    04 BC    70       02F9   426          MOVD    @x(AP), R0              ; R0/R1 = arg
                14    10       02FD   427          BSBB    MTH$DLOG_R8             ; JSB to natural log
      50    C6 AF    64       02FF   428          MULD2   D_INV_LN2_CONS, R0      ; convert to base 2
                04    0303   429          RET                                     ; return - result in R0
                                0304   430
```

```
                     0304   432          .SBTTL  MTH$DLOGDLOG10_R8  - Special DLOG/DLOG10 routines
                     0304   433
                     0304   434 ; Special DLOG/DLOG10 - used by the standard routine, and directly.
                     0304   435 ;
                     0304   436 ; CALLING SEQUENCE:
                     0304   437 ;       save anything needed in R0:R9
                     0304   438 ;       MOVD    ...,R0                  ; input in R0/R1
                     0304   439 ;       JSB     MTH$DLOG10_R8 /MTH$DLOG_R8
                     0304   440 ;       return with result in R0/R1
                     0304   441 ; Note: This routine is written to avoid causing any integer overflows,
                     0304   442 ; floating overflows, or floating underflows or divide by 0 conditions,
                     0304   443 ; whether enabled or not.
                     0304   444 ;
                     0304   445 ; REGISTERS USED:
                     0304   446 ;       R0/R1 - Double floating argument then result
                     0304   447 ;       R2/R3 - scratch
                     0304   448 ;       R0:R5 - POLYD
                     0304   449 ;       R6/R7 - W during POLYD
                     0304   450 ;       R8 - Pointer into D_FHI table
                     0304   451
                     0304   452
                     0304   453
                     0304   454 MTH$DLOG10_R8::                          ; special DLOG10 routine
 58    50    007F 8F   AB  0304   455          BICW3   #^X7F, R0, R8       ; R8 = biased exponent
                 69   15  030A   456          BLEQ    ERR                 ; Error if <= 0
                     030C   457                                          ; Note: ERROR routine depends on user
                     030C   458                                          ; PC being on top of stack, so
                     030C   459                                          ; subroutine call to MTH$DLOG_R8 is
                     030C   460                                          ; not used
                 0D  10  030C   461          BSBB    DLOG_COMMON_R8      ; call common DLOG,DLOG10 routine
      50    AF AF   64  030E   462          MULD    D_DLOG10_E, R0      ; R0/R1 = DLOG10(e) * DLOG(X)
                 05  0312   463          RSB                             ; return
                     0313   464
                     0313   465 MTH$DLOG_R8::                           ; special LOG routine
 58    50    007F 8F   AB  0313   466          BICW3   #^X7F, R0, R8       ; R8 = Biased exponent
                 5A  15  0319   467          BLEQ    ERR                 ; DLOG(X) is not defined for X=<0
                     031B   468 DLOG_COMMON_R8:
       58    4000 8F   A2  031B   469          SUBW    #^X4000, R8         ; R8 = Unbiased exponent
                 56  15  0320   470          BLEQ    NEG_EXP             ; Branch to processing for n=<0
                     0322   471
                     0322   472 ;
                     0322   473 ; Exponent is positive.  N = n - 1 and F = 2f
                     0322   474 ;
                     0322   475
       58    0080 8F   A2  0322   476          SUBW    #^X80, R8           ; R8 = N = n - 1
             50   58   A2  0327   477          SUBW    R8, R0              ; R0/R1 = F = 2f
             53   50   9A  032A   478          MOVZBL  R0, R3              ; R3 = index into MTH$$AB_ALOG table
 53  00000000'GF 43   90  032D   479          MOVB    G^MTH$$AB_ALOG[R3], R3  ; R3 = offset into D_FHI tables
                 3C  19  0335   480          BLSS    LN_1_PLUS           ; Branch to special processing
                     0337   481                                          ;   for F close to 1
                     0337   482
                     0337   483 ;
                     0337   484 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                     0337   485 ;
                     0337   486
          7E   58  6D  0337   487          CVTWD   R8, -(SP)               ; Push N onto the stack
    58   FCC1 CF43  7E  033A   488          MOVAQ   MTH$$AB_D_FHI[R3], R8   ; R8 = Address of FHI
```

```
                54   88   7D  0340   489          MOVQ    (R8)+, R4                ; R4/R5 = FHI
             56 50   54   63  0343   490          SUBD3   R4, R0, R6               ; R6/R7 = F - FHI
                50   54   60  0347   491          ADDD    R4, R0                   ; R0/R1 = F + FHI
                56   50   66  034A   492          DIVD    R0, R6                   ; R6/R7 = Z = (F - FHI)/(F + FHI)
             50 56   56   65  034D   493          MULD3   R6, R6, R0               ; R0/R1 = Z**2
      FF29 CF   05   50   75  0351   494          POLYD   R0, #LOGLEN2-1, LOGTAB2  ; R0/R1 = P(Z**2)
                50   56   64  0357   495          MULD    R6, R0                   ; R0/R1 = Z*P(Z**2)
                                035A   496
                                035A   497 ;
                                035A   498 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                                035A   499 ;
   52   FF59 CF   6E   65  035A   500          MULD3   (SP), D_LN_2_LO, R2      ; R2/R3 = N*LN_2_LO
                52   88   60  0360   501          ADDD    (R8)+, R2                ; R2/R3 = N*LN_2_LO + LN_FHI_LO
                50   52   60  0363   502          ADDD    R2, R0                   ; R0/R1 = B
                                0366   503
                                0366   504 ;
                                0366   505 ; Compute A = N*LN_2_HI + LN_FHI_HI and DLOG(X)
                                0366   506 ;
   52   FF45 CF   8E   65  0366   507          MULD3   (SP)+, D_LN_2_HI, R2     ; R2/R3 = N*LN_2_HI
                52   68   60  036C   508          ADDD    (R8), R2                 ; R2/R3 = A = N*LN_2_HI + LN_FHI_HI
                50   52   60  036F   509          ADDD    R2, R0                   ; R0/R1 = A + B = DLOG(X)
                          05   0372   510          RSB
                                0373   511
                                0373   512 LN_1_PLUS:
                     4F   11   0373   513          BRB     LN_1_PLUS_W
                                0375   514
                0071 31   0375   515 ERR:     BRW     ERROR
                                0378   516
                                0378   517 ;
                                0378   518 ; Exponent is negative.  N = n and F = f
                                0378   519 ;
                                0378   520
                50   58   A2  0378   521 NEG_EXP:SUBW    R8, R0                   ; R0/R1 = F = f
                53   50   9A  037B   522          MOVZBL  R0, R3                   ; R3 = index into MTH$$AB_ALOG table
   53   00000000'GF43   90  037E   523          MOVB    G^MTH$$AB_ALOG[R3], R3   ; R3 = offset into D_FHI tables
                     3C   19  0386   524          BLSS    LN_1_PLUS_W              ; Branch to special processing
                                0388   525                                           ;   for F close to 1
                                0388   526
                                0388   527 ;
                                0388   528 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                                0388   529 ;
                                0388   530
                7E   58   6D  0388   531          CVTWD   R8, -(SP)                ; Push N onto the stack
      58   FC70 CF43   7E  038B   532          MOVAQ   MTH$$AB_D_FHI[R3], R8    ; R8 = Address of FHI
                54   68   7D  0391   533          MOVQ    (R8), R4                 ; R4/R5 = FHI
             56 50   54   63  0394   534          SUBD3   R4, R0, R6               ; R6/R7 = F - FHI
                50   54   60  0398   535          ADDD    R4, R0                   ; R0/R1 = F + FHI
                56   50   66  039B   536          DIVD    R0, R6                   ; R6/R7 = Z = (F - FHI)/(F + FHI)
             50 56   56   65  039E   537          MULD3   R6, R6, R0               ; R0/R1 = Z**2
      FED8 CF   05   50   75  03A2   538          POLYD   R0, #LOGLEN2-1, LOGTAB2  ; R0/R1 = P(Z**2)
                50   56   64  03A8   539          MULD    R6, R0                   ; R0/R1 = Z*P(Z**2)
                                03AB   540
                                03AB   541 ;
                                03AB   542 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                                03AB   543 ;
   52   FF08 CF   6E   65  03AB   544          MULD3   (SP), D_LN_2_LO, R2      ; R2/R3 = N*LN_2_LO
                52   78   60  03B1   545          ADDD    -(R8), R2                ; R2/R3 = N*LN_2_LO + LN_FHI_LO
```

```
                50   52   60   03B4   546          ADDD    R2, R0                  ; R0/R1 = B
                                03B7   547
                                03B7   548 ;
                                03B7   549 ; Compute A = N*LN_2_HI + LN_FHI_HI and DLOG(X)
                                03B7   550 ;
          52    FEF4 CF   8E   65   03B7   551          MULD3   (SP)+, D_LN_2_HI, R2    ; R2/R3 = N*LN_2_HI
                     52   78   62   03BD   552          SUBD    -(R8), R2               ; R2/R3 = A = N*LN_2_HI + LN_FHI_HI
                     50   52   60   03C0   553          ADDD    R2, R0                  ; R0/R1 = A + B = DLOG(X)
                                05   03C3   554          RSB
                                03C4   555
                                03C4   556 ;
                                03C4   557 ; Special logic for F close to 1
                                03C4   558 ;
                                03C4   559
                                03C4   560 LN_1_PLUS_W:
               56   50   08   63   03C4   561          SUBD3   #SD_1, R0, R6           ; R6/R7 = W = F - 1
          FE62 CF   09   56   75   03C8   562          POLYD   R6, #LOGLEN1-1,LOGTAB1  ; R0/R1 = Q(W)
                     50   56   64   03CE   563          MULD    R6, R0                  ; R0/R1 = W*Q(W)
                     54   58   6D   03D1   564          CVTWD   R8, R4                  ; R4/R5 = N
          52    FEDF CF   54   65   03D4   565          MULD3   R4, D_LN_2_LO, R2       ; R2/R3 = N*LN_2_LO
                     50   52   60   03DA   566          ADDD    R2, R0                  ; R0/R1 = N*LN_2_LO + W*Q(W)
                     50   56   60   03DD   567          ADDD    R6, R0                  ; R0/R1 = N*LN_2_LO + LN(F)
               54    FECC CF   64   03E0   568          MULD    D_LN_2_HI, R4           ; R4/R5 = N*LN_2_HI
                     50   54   60   03E5   569          ADDD    R4, R0                  ; R0/R1 = DLOG(X)
                                05   03E8   570          RSB
                                03E9   571
                                03E9   572 ;
                                03E9   573 ; X =< 0.0, signal error
                                03E9   574 ;
                          6E    DD   03E9   575 ERROR:  PUSHL   (SP)                    ; return PC from JSB routine
                     7E   00'8F    9A   03EB   576          MOVZBL  #MTH$K_LOGZERNEG, -(SP) ; condition value
               50    01   0F    79   03EF   577          ASHQ    #15, #1, R0             ; R0 = result = reserved operand -0.0
                                03F3   578                                              ; goes to signal mechanism vector
                                03F3   579                                              ; (CHF$L_MCH_R0/R1) so error handler
                                03F3   580                                              ; can modify the result.
00000000'GF     02   FB   03F3   581          CALLS   #2, G^MTH$$SIGNAL               ; signal error and use real user's PC
                                03FA   582                                              ; independent of CALL vs JSB
                                05   03FA   583          RSB                             ; return - R0 restored from
                                03FB   584                                              ; CHF$L_MCH_R0/R1
                                03FB   585
                                03FB   586
                                03FB   587
                                03FB   588          .END
```

```
ACMASK          = 000041FC
DLOG_COMMON_R8    0000031B R      01
D_DLOG10_E        000002C0 R      01
D_INV_LN2_CONS    000002C8 R      01
D_LN_2_HI         000002B0 R      01
D_LN_2_LO         000002B8 R      01
ERR               00000375 R      01
ERROR             000003E9 R      01
LN_1_PLUS         00000373 R      01
LN_1_PLUS_W       000003C4 R      01
LOGLEN1         = 0000000A
LOGLEN2         = 00000006
LOGTAB1           00000230 R      01
LOGTAB2           00000280 R      01
LONG            = 00000004
MTH$$AB_ALOG      ********    X   00
MTH$$AB_D_FHI     00000000 RG     01
MTH$$JACKET_HND   ********    X   01
MTH$$SIGNAL       ********    X   00
MTH$DLOG          000002D0 RG     01
MTH$DLOG10        000002E0 RG     01
MTH$DLOG10_R8     00000304 RG     01
MTH$DLOG2         000002F0 RG     01
MTH$DLOG_R8       00000313 RG     01
MTH$K_LOGZERNEG   ********    X   00
NEG_EXP           00000378 R      01
SD_T            = 00004080
X               = 00000004
```

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

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_MTH$CODE	000003FB (1019.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.10	00:00:00.63
Command processing	119	00:00:00.63	00:00:05.45
Pass 1	104	00:00:01.66	00:00:04.23
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	115	00:00:01.39	00:00:04.17
Symbol table output	4	00:00:00.03	00:00:00.04
Psect synopsis output	2	00:00:00.02	00:00:00.09
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	377	00:00:03.85	00:00:14.63

The working set limit was 1050 pages.
9323 bytes (19 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 28 non-local and 0 local symbols.

648 source lines were read in Pass 1, producing 18 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

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

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	0

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:MTHDLOG/OBJ=OBJ$:MTHDLOG 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