```
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
```

FILEID**MTHGLOG

```
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG  LL           OOOOOO      GGGGGGGG
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG  LL           OOOOOO      GGGGGGGG
MMMM  MMMM      TT      HH      HH  GG          LL         OO      OO  GG
MMMM  MMMM      TT      HH      HH  GG          LL         OO      OO  GG
MM  MM  MM      TT      HH      HH  GG          LL         OO      OO  GG
MM  MM  MM      TT      HH      HH  GG          LL         OO      OO  GG
MM      MM      TT      HHHHHHHHHH  GG          LL         OO      OO  GG
MM      MM      TT      HHHHHHHHHH  GG          LL         OO      OO  GG
MM      MM      TT      HH      HH  GG  GGGGGG  LL         OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  GG  GGGGGG  LL         OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  GG      GG  LL         OO      OO  GG      GG      ....
MM      MM      TT      HH      HH  GG      GG  LL         OO      OO  GG      GG      ....
MM      MM      TT      HH      HH    GGGGGG    LLLLLLLLLL   OOOOOO      GGGGGG        ....
MM      MM      TT      HH      HH    GGGGGG    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$GLOG        ; Floating Point Natural and Common
0000     2                                  ; Logarithm Functions (GLOG,GLOG10)
0000     3          .IDENT /2-005/          ; File: MTHGLOG.MAR     PDG2005
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$GLOG  and  MTH$GLOG10  are functions which return the G floating natural
0000    34 ; or common logarithm of their G floating point argument. The call is standard
0000    35 ; call-by-reference.  MTH$GLOG_R8 and MTH$GLOG10_R8 are special routines which
0000    36 ; are the same as MTH$GLOG and  MTH$GLOG10  except  a  faster non-standard JSB
0000    37 ; call is used with the argument in R0 and no registers are saved.
0000    38 ;
0000    39 ;--
0000    40 ;
0000    41 ; VERSION: 1
0000    42 ;
0000    43 ; HISTORY:
0000    44 ; AUTHOR:
0000    45 ;       Steven B. Lionel, 18-Jan-1979
0000    46 ;
0000    47 ; MODIFIED BY:
0000    48 ;
0000    49 ;
0000    50 ; VERSION: 2
0000    51 ;
0000    52 ; HISTORY:
0000    53 ; AUTHOR:
0000    54 ;       Bob Hanek, 18-Jun-1981
0000    55 ;
0000    56 ;
```

```
0000     58          .SBTTL  HISTORY ; Detailed Current Edit History
0000     59
0000     60
0000     61 ; ALGORITHMIC DIFFERENCE FROM FP-11C ROUTINE:
0000     62 ; \\ D used in comparison, FP-11C has no G \\
0000     63 ;       1. Uses POLYD so greater accuracy.
0000     64 ;
0000     65 ; Edit History for Version 1 of MTH$GLOG
0000     66 ;
0000     67 ; 1-001 - Adapted from MTH$DLOG version 1-010.  SBL 18-Jan-79
0000     68 ;
0000     69 ;
0000     70 ; Edit History for Version 2 of MTH$GLOG
0000     71 ;
0000     72 ; 2-001 - Added MTH$GLOG2.  RNH 08-Aug-1981
0000     73 ; 2-002 - Correct entry logic in JSB entry points.  Use G^ addressing for
0000     74 ;         externals.  SBL 24-Aug-1981
0000     75 ; 2-003 - Changed MTH$$AB_ALOG to MTH$$AB_ALOG_V  RNH 29-Sep-81
0000     76 ; 2-004 - Eliminated symbolic short literals.  RNH 15-Oct-81
0000     77 ; 2-005 - Changed G_FHI to the global symbol MTH$$AB_G_FHI.  PDG 3-Nov-81
```

```
                     0000     79          .SBTTL  DECLARATIONS    ; Declarative Part of Module
                     0000     80
                     0000     81 ;
                     0000     82 ; INCLUDE FILES:         MTHJACKET.MAR
                     0000     83 ;
                     0000     84 ; EXTERNAL SYMBOLS:
                     0000     85          .DSABL  GBL
                     0000     86          .EXTRN  MTH$K_LOGZERNEG         ; Error code
                     0000     87          .EXTRN  MTH$$SIGNAL             ; Math signal routine
                     0000     88          .EXTRN  MTH$$AB_ALOG_V          ; Table of byte offsets
                     0000     89
                     0000     90 ; EQUATED SYMBOLS:
                     0000     91
                000041FC     92          ACMASK = ^M<IV, R2, R3, R4, R5, R6, R7, R8>
                     0000     93                                          ; register save mask and IV enable
                     0000     94
                     0000     95 ;
                     0000     96 ; MACROS:        none
                     0000     97 ;
                     0000     98 ; PSECT DECLARATIONS:
                     0000     99
                00000000    100          .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                     0000    101                                          ; program section for math routines
                     0000    102 ;
                     0000    103 ; OWN STORAGE:  none
                     0000    104 ;
                     0000    105 ; CONSTANTS:
                     0000    106
                     0000    107 ;
                     0000    108 ;
                     0000    109 ;   The G_FHI table is accessed by an index obtained from the MTH$$AB_ALOG_V
                     0000    110 ;   table.  The MTH$$AB_ALOG_V table is  located  in  MTHALOG.MAR.  Indices
                     0000    111 ;   between  0  and  13  inclusive  are  used  to access entries 0 through 13
                     0000    112 ;   respectively.  For  these  indecies,  the  first  three  items  of  the
                     0000    113 ;   corresponding entry are  FHI,  LN_FHI_LO  and  LN_FHI_HI .  The last two
                     0000    114 ;   items  for  these  entries  are  not  used.  Indices  between  14 and 27
                     0000    115 ;   inclusive access entries 13 through 0 respectively.  For  these indecies,
                     0000    116 ;   the last three items in the corresponding entry are LN_FHI_HI, LN_FHI_LO
                     0000    117 ;   and FHI.  The first two items for these entries are not used.
                     0000    118 ;
                     0000    119
                     0000    120 MTH$$AB_G_FHI::
                     0000    121 ; Entry  0
00000000 A9F2401D    0000    122          .QUAD   ^X00000000A9F2401D      ; .18539905548095703E+01
CF83A5D1 989D3E73    0008    123          .QUAD   ^XCF83A5D1989D3E73      ; .18250342005397692E-07
00008F13 C1404003    0010    124          .QUAD   ^X00008F13C1404003      ; .61734035438712453E+00
70214E30 94D23E73    0018    125          .QUAD   ^X70214E3094D23E73      ; .18236538401006972E-07
0000C000 42934001    0020    126          .QUAD   ^X0000C00042934001      ; .53937709331512451E+00
                     0028    127 ; Entry  1
00006000 E3A84019    0028    128          .QUAD   ^X00006000E3A84019      ; .16180804967880249E+01
47A62B6F FA08BE81    0030    129          .QUAD   ^X47A62B6FFA08BE81      ;-.33484189136366529E-07
0000616E CCA53FFE    0038    130          .QUAD   ^X0000616ECCA53FFE      ; .48124060167901916E+00
0174A8C5 FB8CBE81    0040    131          .QUAD   ^X0174A8C5FB8CBE81      ;-.33495230674590973E-07
0000E000 C6C94003    0048    132          .QUAD   ^X0000E000C6C94003      ; .61801618337631226E+00
                     0050    133 ; Entry  2
00008000 4D1A4017    0050    134          .QUAD   ^X000080004D1A4017      ; .14563241004943848E+01
34C26ADE C1B23E60    0058    135          .QUAD   ^X34C26ADEC1B23E60      ; .78029132840604787E-08
```

```
0000F158 0EFF3FF8  0060   136          .QUAD   ^X0000F1580EFF3FF8      ; .3759155136940535 2E+00
B9B690B9 D4813E60  0068   137          .QUAD   ^XB9B690B9D4813E60      ; .7837126967543960 7E-08
00002000 F91F4005  0070   138          .QUAD   ^X00002000F91F4005      ; .6866603493690490 7E+00
                   0078   139 ; Entry  3
0000A000 75A34015  0078   140          .QUAD   ^X0000A00075A34015      ; .1341220498085022 0E+01
81C3B006 52B4BE4C  0080   141          .QUAD   ^X81C3B00652B4BE4C      ;-.3297239259579653 4E-08
0000DC82 CA033FF2  0088   142          .QUAD   ^X0000DC82CA033FF2      ; .2935800221821409 7E+00
69CA531D 6247BE4C  0090   143          .QUAD   ^X69CA531D6247BE4C      ;-.3304320949602087 2E-08
0000A000 DBDE4007  0098   144          .QUAD   ^X0000A000DBDE4007      ; .7455895543098449 7E+00
                   00A0   145 ; Entry  4
00004000 23B44014  00A0   146          .QUAD   ^X0000400023B44014      ; .1258716821670532 2E+01
59ADA334 CFCA3E5E  00A8   147          .QUAD   ^X59ADA334CFCA3E5E      ; .7173904625963530 6E-08
00004C2D 73AE3FED  00B0   148          .QUAD   ^X00004C2D73AE3FED      ; .2300927993728691 9E+00
B88B5562 C8593E5E  00B8   149          .QUAD   ^XB88B5562C8593E5E      ; .7167135626451720 6E-08
00002000 6C374009  00C0   150          .QUAD   ^X000020006C374009      ; .7944598793983459 5E+00
                   00C8   151 ; Entry  5
00004000 317A4013  00C8   152          .QUAD   ^X00004000317A4013      ; .1199579477310180 7E+01
8CE2216E 5F503E73  00D0   153          .QUAD   ^X8CE2216E5F503E73      ; .1804187562858479 1E-07
0000BC97 4AD33FE7  00D8   154          .QUAD   ^X0000BC974AD33FE7      ; .1819710417603346 3E+00
F600D2D6 5FA23E73  00E0   155          .QUAD   ^XF600D2D65FA23E73      ; .1804305076678564 9E-07
00006000 AD0F400A  00E8   156          .QUAD   ^X00006000AD0F400A      ; .8336254954338073 7E+00
                   00F0   157 ; Entry  6
0000C000 7FF44012  00F0   158          .QUAD   ^X0000C0007FF44012      ; .1156239271163940 4E+01
54D6FF1B 54DF3E7A  00F8   159          .QUAD   ^X54D6FF1B54DF3E7A      ; .2452316034166975 0E-07
0000ECE2 95043FE2  0100   160          .QUAD   ^X0000ECE295043FE2      ; .1451727062849386 1E+00
B698EB39 550F3E7A  0108   161          .QUAD   ^XB698EB39550F3E7A      ; .2452384135906107 2E-07
00000000 AD0A400B  0110   162          .QUAD   ^X00000000AD0A400B      ; .8648729324340820 3E+00
                   0118   163 ; Entry  7
00008000 F8314011  0118   164          .QUAD   ^X00008000F8314011      ; .1123094081878662 1E+01
7DC6AF4B D72E3E61  0120   165          .QUAD   ^X7DC6AF4BD72E3E61      ; .8307656321062892 3E-08
0000137D B7E83FDD  0128   166          .QUAD   ^X0000137DB7E83FDD      ; .1160874412134944 6E+00
E41C4BBB D54D3E61  0130   167          .QUAD   ^XE41C4BBBD54D3E61      ; .8304235847132730 0E-08
0000A000 7E22400C  0138   168          .QUAD   ^X0000A0007E22400C      ; .8903973698616027 8E+00
                   0140   169 ; Entry  8
00002000 8B674011  0140   170          .QUAD   ^X000020008B674011      ; .1096533894538879 4E+01
181167D5 31D6BE76  0148   171          .QUAD   ^X181167D531D6BE76      ;-.2067040448985304 9E-07
000043D7 976B3FD7  0150   172          .QUAD   ^X000043D7976B3FD7      ; .9215422064062295 2E-01
FA16278D 2F4DBE76  0158   173          .QUAD   ^XFA16278D2F4DBE76      ;-.2066117807700813 9E-07
00002000 2ED0400D  0160   174          .QUAD   ^X000020002ED0400D      ; .9119644761085510 3E+00
                   0168   175 ; Entry  9
00004000 36564011  0168   176          .QUAD   ^X0000400036564011      ; .1075765848159790 0E+01
EBD925C6 6CAA3E81  0170   177          .QUAD   ^XEBD925C66CAA3E81      ; .3245560684395479 3E-07
0000EEC2 B2463FD2  0178   178          .QUAD   ^X0000EEC2B2463FD2      ; .7303279236839443 9E-01
CF5A4740 6B433E81  0180   179          .QUAD   ^XCF5A47406B433E81      ; .3244540716686605 1E-07
00006000 BF0A400D  0188   180          .QUAD   ^X00006000BF0A400D      ; .9295703768730163 6E+00
                   0190   181 ; Entry 10
0000E000 F69B4010  0190   182          .QUAD   ^X0000E000F69B4010      ; .1060207247734069 8E+01
04DF36B7 35D33E77  0198   183          .QUAD   ^X04DF36B735D33E77      ; .2161623362056486 6E-07
0000339D EF0B3FCD  01A0   184          .QUAD   ^X0000339DEF0B3FCD      ; .5846438412709176 2E-01
FC0C3872 36323E77  01A8   185          .QUAD   ^XFC0C387236323E77      ; .2161758374803248 9E-07
0000A000 2ECA400E  01B0   186          .QUAD   ^X0000A0002ECA400E      ; .9432118535041809 1E+00
                   01B8   187 ; Entry 11
00008000 CA844010  01B8   188          .QUAD   ^X00008000CA844010      ; .1049442768096923 8E+01
BD3E1C71 5686BE82  01C0   189          .QUAD   ^XBD3E1C715686BE82      ;-.3415715707991671 1E-07
00006D58 B5733FC8  01C8   190          .QUAD   ^X00006D58B5733FC8      ; .4825936040651868 1E-01
4870F892 5619BE82  01D0   191          .QUAD   ^X4870F8925619BE82      ;-.3415408318068389 3E-07
00000000 7E0C400E  01D8   192          .QUAD   ^X000000007E0C400E      ; .9528865814208984 4E+00
```

```
                        01E0   193 ; Entry 12
00002000 A7094010       01E0   194          .QUAD   ^X00002000A7094010      ; .10407801866531372E+01
35CB6848 0BB93E65       01E8   195          .QUAD   ^X35CB68480BB93E65      ; .98002133153715869E-08
0000D534 77063FC4       01F0   196          .QUAD   ^X0000D53477063FC4      ; .39970601583263488E-01
9A9B90CD 071C3E65       01F8   197          .QUAD   ^X9A9B90CD071C3E65      ; .97918229303478694E-08
0000C000 BF04400E       0200   198          .QUAD   ^X0000C000BF04400E      ; .96081769466400146E+00
                        0208   199 ; Entry 13
00002000 8DDD4010       0208   200          .QUAD   ^X000020008DDD4010      ; .10346347093582153E+01
EBA761A0 A9EC3E68       0210   201          .QUAD   ^XEBA761A0A9EC3E68      ; .11484959695179258E-07
00003AD1 6ECB3FC1       0218   202          .QUAD   ^X00003AD16ECB3FC1      ; .34048415117013064E-01
450394DF A64D3E68       0220   203          .QUAD   ^X450394DFA64D3E68      ; .11478374386313017E-07
00004000 EDC5400E       0228   204          .QUAD   ^X00004000EDC5400E      ; .96652472019195557E+00
                        0230   205
                        0230   206 ;
                        0230   207 ;  Polynomial constants tables
                        0230   208 ;
                        0230   209
                        0230   210
                        0230   211 LOGTAB1:                                ; Constants for q(z).  Generated using
                        0230   212                                         ; eq. 6.3.10 of Hart et. al. (sin(2a)
                        0230   213                                         ; = 1/32)
A8981E57 81CD3FDC       0230   214          .QUAD   ^XA8981E5781CD3FDC      ; C8 =   0.11135560980588577
38EFC0D0 0802BFE0       0238   215          .QUAD   ^X38EFC0D00802BFE0      ; C7 =  -0.12524446882930060
C9769148 49223FE2       0240   216          .QUAD   ^XC976914849223FE2      ; C6 =   0.14285690397225509
BBAC9487 5553BFE5       0248   217          .QUAD   ^XBBAC94875553BFE5      ; C5 =  -0.16666645767642529
B92699D1 99993FE9       0250   218          .QUAD   ^XB92699D199993FE9      ; C4 =   0.20000000010208757
0A540014 0000BFF0       0258   219          .QUAD   ^X0A5400140000BFF0      ; C3 =  -0.25000000007290635
54155555 55553FF5       0260   220          .QUAD   ^X5415555555553FF5      ; C2 =   0.33333333333331555
FF60FFFF FFFFBFFF       0268   221          .QUAD   ^XFF60FFFFFFFFBFFF      ; C1 =  -0.49999999999999112
00000000 00000000       0270   222          .QUAD   ^X0000000000000000      ; C0 =   0.00000000000000000
         00000009       0278   223 LOGLEN1 = .-LOGTAB1/8                   ; no. of floating point entries
                        0278   224
                        0278   225
                        0278   226 LOGTAB2:                                ; Constants for p(z*z). Generated using
                        0278   227                                         ; eq. 6.3.11 of Hart et. al. (sin(2a) =
                        0278   228                                         ; (b - 1)/(b + 1) where b = 2**(1/7))
B117401D 6E163FE7       0278   229          .QUAD   ^XB117401D6E163FE7      ; C5 =  0.18304708605445l497
0BA587C0 71A73FEC       0280   230          .QUAD   ^X0BA587C071A73FEC      ; C4 =  0.22221845749308247l
C30B9839 49243FF2       0288   231          .QUAD   ^XC30B983949243FF2      ; C3 =  0.28571429124626551l
839E9998 99993FF9       0290   232          .QUAD   ^X839E999899993FF9      ; C2 =  0.39999999996049627
55605555 55554005       0298   233          .QUAD   ^X5560555555554005      ; C1 =  0.66666666666667851
00000000 00004020       02A0   234          .QUAD   ^X0000000000004020      ; C0 =  2.00000000000000000
         00000006       02A8   235 LOGLEN2 = .-LOGTAB2/8
                        02A8   236
                        02A8   237 ;+  The "16" in the constants is used to shift the unbiased exponent
                        02A8   238 ;   right 4 places so that the rightmost bit is at bit 0.
                        02A8   239 ;-
                        02A8   240
                        02A8   241 G_LN_2_HI:
2800FEF6 2E423FC6       02A8   242          .QUAD   ^X2800FEF62E423FC6      ; (Hi 42 bits of ln2)/16
                        02B0   243 G_LN_2_LO:
F1DAD5E4 47BC3DA0       02B0   244          .QUAD   ^XF1DAD5E447BC3DA0      ; (Low bits of ln2)/16
                        02B8   245 G_GLOG10_E:                              ; LOG10(e)
         CB7B 3FFB      02B8   246          .WORD   ^O037773,^O145573
         E50E 1526      02BC   247          .WORD   ^O012446,^O162416
                        02C0   248 G_INV_LN2_CONS:
82FE652B 15474017       02C0   249          .QUAD   ^X82FE652B15474017
```

02C8 250

```
                        02C8    252          .SBTTL  MTH$GLOG  - Standard G-Floating LOG
                        02C8    253
                        02C8    254
                        02C8    255 ;++
                        02C8    256 ; FUNCTIONAL DESCRIPTION:
                        02C8    257 ;
                        02C8    258 ; GLOG  - single precision floating point function
                        02C8    259 ;
                        02C8    260 ; GLOG(X) is computed using the following approximation technique:
                        02C8    261 ;
                        02C8    262 ;       If X =< 0, error.  Otherwise
                        02C8    263 ;
                        02C8    264 ;       Let X = f * (2**n), where 1/2 <= f < 1
                        02C8    265 ;
                        02C8    266 ;       If n is greater than or equal to 1 than
                        02C8    267 ;             set N = n - 1 and F = 2*f.
                        02C8    268 ;          Else
                        02C8    269 ;             set N = n and F = f.
                        02C8    270 ;
                        02C8    271 ;       Then  ln(x) = N*ln2 + ln(F)
                        02C8    272 ;
                        02C8    273 ;       If !F - 1! < 2**-5 then
                        02C8    274 ;             ln(F) =  W + W*P(W), where W = F - 1 and P
                        02C8    275 ;             is a polynomial of degree 8.
                        02C8    276 ;          Else
                        02C8    277 ;             ln(F) = ln(FHI) + Z*Q(Z*Z), where FHI is ob-
                        02C8    278 ;             tained by table look-up, Q is a polynomial of
                        02C8    279 ;             degree 5 and Z = (F - FHI)/(F + FHI)
                        02C8    280 ;
                        02C8    281 ;       NOTE: The quantities ln(FHI) and ln2 are used in the above
                        02C8    282 ;             equations in two parts - a high part (containing the
                        02C8    283 ;             high order bits) and a low part (containing the low
                        02C8    284 ;             order bits.  In the code the high and low parts of the
                        02C8    285 ;             constants are indicated by a _HI and _LO suffix respec-
                        02C8    286 ;             tively.  The values were chosen such that N*LN_2_HI +
                        02C8    287 ;             LN_FHI_HI is exactly representable.
                        02C8    288 ;
                        02C8    289 ; CALLING SEQUENCE:
                        02C8    290 ;
                        02C8    291 ;       logarithm.wg.v = MTH$GLOG(x.rg.r)
                        02C8    292 ;
                        02C8    293 ; INPUT PARAMETERS:
                        02C8    294
00000004                02C8    295          LONG = 4                        ; define longword multiplier
00000004                02C8    296          x = 1 * LONG                    ; Contents of x is the argument
                        02C8    297
                        02C8    298 ; IMPLICIT INPUTS:      none
                        02C8    299 ;
                        02C8    300 ; OUTPUT PARAMETERS:
                        02C8    301 ;
                        02C8    302 ;       VALUE:  G floating logarithm of the argument
                        02C8    303 ;
                        02C8    304 ; IMPLICIT OUTPUTS:     none
                        02C8    305 ;
                        02C8    306 ; COMPLETION CODES:     none
                        02C8    307 ;
                        02C8    308 ; SIDE EFFECTS:
```

```
                                02C8    309 ;
                                02C8    310 ; Signals: MTH$_LOGZERNEG if !X! =< 0.0 with reserved operand in R0/R1
                                02C8    311 ; (copied to the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                                02C8    312 ; Associated message is: "LOGARITHM OF ZERO OR NEGATIVE VALUE". Result is
                                02C8    313 ; reserved operand -0.0 unless a user supplied (or any) error handler changes
                                02C8    314 ; CHF$L_MCH_R0/R1.
                                02C8    315 ;
                                02C8    316 ; NOTE: This procedure disables floating point underflow, enables integer
                                02C8    317 ; overflow, causes no floating overflow or other arithmetic traps, and
                                02C8    318 ; preserves enables across the call.
                                02C8    319 ;
                                02C8    320 ;---
                                02C8    321
                                02C8    322
                         41FC   02C8    323         .ENTRY  MTH$GLOG, ACMASK        ; standard call-by-reference entry
                                02CA    324                                         ; disable DV (and FU), enable IV
                                02CA    325         MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                02CA
    6D  00000000'GF      9E     02CA                 MOVAB   G^MTH$$JACKET_HND, (FP)
                                02D1                                                ; set handler address to jacket
                                02D1                                                ; handler
                                02D1
                                02D1    326                                         ; in case of an error in special JSB
                                02D1    327                                         ; routine
        50    04 BC 50FD        02D1    328         MOVG    @x(AP), R0              ; R0/R1 = arg
                    39   10     02D6    329         BSBB    MTH$GLOG_R8             ; call special GLOG routine
                         04     02D8    330         RET                             ; return - result in R0/R1
                                02D9    331
```

```
                                02D9    333          .SBTTL  MTH$GLOG10  - Standard G Floating Common logarithm
                                02D9    334
                                02D9    335 ;++
                                02D9    336 ; FUNCTIONAL DESCRIPTION:
                                02D9    337 ;
                                02D9    338 ; GLOG10  - G floating point function
                                02D9    339 ;
                                02D9    340 ; GLOG10(X) is computed as GLOG10(E) * GLOG(X).
                                02D9    341 ;
                                02D9    342 ; See description of MTH$GLOG
                                02D9    343 ;
                                02D9    344 ; CALLING SEQUENCE:
                                02D9    345 ;
                                02D9    346 ;       logarithm_base_10.wg.v = MTH$GLOG10(x.rg.r)
                                02D9    347 ;
                                02D9    348 ; INPUT PARAMETERS:
                                02D9    349
                    00000004    02D9    350          LONG = 4                        ; define longword multiplier
                    00000004    02D9    351          x = 1 * LONG                    ; Contents of x is the argument
                                02D9    352
                                02D9    353 ;
                                02D9    354 ; SIDE EFFECTS: See description of MTH$GLOG
                                02D9    355 ;
                                02D9    356 ;--
                                02D9    357
                                02D9    358
                        41FC    02D9    359          .ENTRY  MTH$GLOG10, ACMASK      ; standard call-by-reference entry
                                02DB    360                                          ; disable DV (and FU), enable IV
                                02DB    361          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                02DB
            6D  00000000'GF      9E 02DB                       MOVAB   G^MTH$$JACKET_HND, (FP)
                                02E2                                                 ; set handler address to jacket
                                02E2                                                 ; handler
                                02E2
                                02E2    362                                          ; in case of an error in special JSB
                                02E2    363                                          ; routine
                50  04 BC 50FD  02E2    364          MOVG    @x(AP), R0              ; R0/R1 = arg
                     17   10    02E7    365          BSBB    MTH$GLOG10_R8           ; call special GLOG10 routine
                          04    02E9    366          RET                             ; return - result in R0/R1
                                02EA    367
                                02EA    368
```

```
                                02EA   370          .SBTTL  MTH$GLOG2  - Standard G Floating Common logarithm
                                02EA   371
                                02EA   372 ;++
                                02EA   373 ; FUNCTIONAL DESCRIPTION:
                                02EA   374 ;
                                02EA   375 ; GLOG2  - G floating point function
                                02EA   376 ;
                                02EA   377 ; GLOG2(X) is computed as GLOG2(E) * GLOG(X).
                                02EA   378 ;
                                02EA   379 ; See description of MTH$GLOG
                                02EA   380 ;
                                02EA   381 ; CALLING SEQUENCE:
                                02EA   382 ;
                                02EA   383 ;       logarithm_base_2.wg.v = MTH$GLOG2(x.rg.r)
                                02EA   384 ;
                                02EA   385 ; INPUT PARAMETERS:
                                02EA   386
                    00000004    02EA   387          LONG = 4                        ; define longword multiplier
                    00000004    02EA   388          x = 1 * LONG                    ; Contents of x is the argument
                                02EA   389
                                02EA   390 ;
                                02EA   391 ; SIDE EFFECTS: See description of MTH$GLOG
                                02EA   392 ;
                                02EA   393 ;--
                                02EA   394
                                02EA   395
                        41FC    02EA   396          .ENTRY  MTH$GLOG2, ACMASK       ; standard call-by-reference entry
                                02EC   397                                          ; disable DV (and FU), enable IV
                                02EC   398          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                02EC
        6D  00000000'GF    9E   02EC               MOVAB   G^MTH$$JACKET_HND, (FP)
                                02F3                                                ; set handler address to jacket
                                02F3                                                ; handler
                                02F3
                                02F3   399                                          ; in case of an error in special JSB
                                02F3   400                                          ; routine
        50    04 BC 50FD        02F3   401          MOVG    @x(AP), R0              ; R0/R1 = arg
                17   10         02F8   402          BSBB    MTH$GLOG_R8             ; jump to natural log
        50    C2 AF 44FD        02FA   403          MULG2   G_INV_LN2_CONS, R0      ; convert to log base 2
                        04      02FF   404          RET                             ; return - result in R0/R1
                                0300   405
                                0300   406
```

```
                          0300   408          .SBTTL  MTH$GLOGGLOG10_R8  - Special GLOG/GLOG10 routines
                          0300   409
                          0300   410 ; Special GLOG/GLOG10 - used by the standard routine, and directly.
                          0300   411 ;
                          0300   412 ; CALLING SEQUENCE:
                          0300   413 ;       save anything needed in R0:R9
                          0300   414 ;       MOVG    ...,R0                  ; input in R0/R1
                          0300   415 ;       JSB     MTH$GLOG10_R8 /MTH$GLOG_R8
                          0300   416 ;       return with result in R0/R1
                          0300   417 ; Note: This routine is written to avoid causing any integer overflows,
                          0300   418 ; floating overflows, or floating underflows or divide by 0 conditions,
                          0300   419 ; whether enabled or not.
                          0300   420 ;
                          0300   421 ; REGISTERS USED:
                          0300   422 ;       R0/R1 - G floating argument then result
                          0300   423 ;       R2/R3 - scratch
                          0300   424 ;       R0:R5 - POLYG
                          0300   425 ;       R6/R7 - W during POLYG
                          0300   426 ;       R8 - Pointer into G_FHI table
                          0300   427
                          0300   428
                          0300   429
                          0300   430 MTH$GLOG10_R8::                          ; special GLOG10 routine
 58    50    0F    AB     0300   431          BICW3   #^XF, R0, R8            ; R8 = Biased exponent
             08    15     0304   432          BLEQ    ERR                     ; GLOG(X) is not defined for X=<0
                          0306   433                                          ; user PC on top of stack
                          0306   434                                          ; Note: ERROR routine depends on user
                          0306   435                                          ; PC being on top of stack, so
                          0306   436                                          ; subroutine call to MTH$DLOG_R8 is not
                          0306   437                                          ; used.
             0F    10     0306   438          BSBB    GLOG_COMMON_R8          ; call common GLOG/GLOG10 routine
   50    AC AF 44FD       0308   439          MULG2   G_GLOG10_E, R0          ; R0/R1 = GLOG10(e) * GLOG(X)
                 05       030D   440          RSB                             ; return
                          030E   441
               010E  31   030E   442 ERR:     BRW     ERROR
                          0311   443
                          0311   444 MTH$GLOG_R8::                           ; special LOG routine
 58    50    0F    AB     0311   445          BICW3   #^XF, R0, R8            ; R8 = Biased exponent
             F7    15     0315   446          BLEQ    ERR                     ; GLOG(X) is not defined for X=<0
                          0317   447 GLOG_COMMON_R8:
 58    4000  8F    A2     0317   448          SUBW    #^X4000, R8             ; R8 = Unbiased exponent
             6C    15     031C   449          BLEQ    NEG_EXP                 ; Branch to processing for n=<0
                          031E   450
                          031E   451 ;
                          031E   452 ; Exponent is positive.  N = n - 1 and F = 2f
                          031E   453 ;
                          031E   454
       58    10    A2     031E   455          SUBW    #^X10, R8               ; R8 = N = n - 1
       50    58    A2     0321   456          SUBW    R8, R0                  ; R0/R1 = F = 2f
 53    50    03    9C     0324   457          ROTL    #3, R0, R3              ; R3 = index into MTH$$AB_ALOG_V table
53  FFFFFF00  8F    CA     0328   458          BICL    #-256, R3               ;    = lo exp bit and 1st 7 fract bits
52  00000000'GF   DE     032F   459          MOVAL   G^MTH$$AB_ALOG_V, R2    ; R2 = Address of RTL vector entry
      52    62    C0     0336   460          ADDL    (R2), R2                ; R2 = Address of MTH$AB_ALOG table
        53  6243  90     0339   461          MOVB    (R2)[R3], R3            ; R3 = offset into G_FHI tables
              49  19     033D   462          BLSS    LN_1_PLUS               ; Branch to special processing
                          033F   463                                          ;   for F close to 1
                          033F   464
```

```
                     033F   465 ;
                     033F   466 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                     033F   467 ;
                     033F   468
          7E  58 4DFD  033F   469          CVTWG   R8, -(SP)                ; Push N onto the stack
    58  FCB8 CF43  7E  0343   470          MOVAQ   MTH$$AB_G_FHI[R3], R8    ; R8 = Address of FHI
        54   88  7D  0349   471          MOVQ    (R8)+, R4                ; R4/R5 = FHI
    56  50   54 43FD  034C   472          SUBG3   R4, R0, R6               ; R6/R7 = F - FHI
        50   54 40FD  0351   473          ADDG2   R4, R0                   ; R0/R1 = F + FHI
        56   50 46FD  0355   474          DIVG2   R0, R6                   ; R6/R7 = Z = (F - FHI)/(F + FHI)
    50  56   56 45FD  0359   475          MULG3   R6, R6, R0               ; R0/R1 = Z**2
FF13 CF 05   50 55FD  035E   476          POLYG   R0, #LOGLEN2-1, LOGTAB2  ; R0/R1 = P(Z**2)
        50   56 44FD  0365   477          MULG2   R6, R0                   ; R0/R1 = Z*P(Z**2)
                     0369   478
                     0369   479 ;
                     0369   480 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                     0369   481 ;
52  FF41 CF  6E 45FD  0369   482          MULG3   (SP), G_LN_2_LO, R2      ; R2/R3 = N*LN_2_LO
        52   88 40FD  0370   483          ADDG2   (R8)+, R2                ; R2/R3 = N*LN_2_LO + LN_FHI_LO
        50   52 40FD  0374   484          ADDG2   R2, R0                   ; R0/R1 = B
                     0378   485
                     0378   486 ;
                     0378   487 ; Compute A = N*LN_2_HI + LN_FHI_HI and GLOG(X)
                     0378   488 ;
52  FF2A CF  8E 45FD  0378   489          MULG3   (SP)+, G_LN_2_HI, R2     ; R2/R3 = N*LN_2_HI
        52   68 40FD  037F   490          ADDG2   (R8), R2                 ; R2/R3 = A = N*LN_2_HI + LN_FHI_HI
        50   52 40FD  0383   491          ADDG2   R2, R0                   ; R0/R1 = A + B = GLOG(X)
             05      0387   492          RSB
                     0388   493
                     0388   494
                     0388   495 LN_1_PLUS:
        67   11      0388   496          BRB     LN_1_PLUS_W
                     038A   497
                     038A   498
                     038A   499 ;
                     038A   500 ; Exponent is negative.  N = n and F = f
                     038A   501 ;
                     038A   502
     50   58   A2  038A   503 NEG_EXP:SUBW    R8, R0                   ; R0/R1 = F = f
53   50   03   9C  038D   504          ROTL    #3, R0, R3               ; R3 = index into MTH$$AB_ALOG table
53 FFFFFF00 8F   CA  0391   505          BICL    #-256, R3                ;    = lo exp bit and 1st 7 fract bits
52 00000000'GF   DE  0398   506          MOVAL   G^MTH$$AB_ALOG_V, R2     ; R2 = Address of RTL vector entry
     52   62   C0  039F   507          ADDL    (R2), R2                 ; R2 = Address of MTH$AB_ALOG table
   53   6243   90  03A2   508          MOVB    (R2)[R3], R3             ; R3 = offset into G_FHI tables
        49   19  03A6   509          BLSS    LN_1_PLUS_W              ; Branch to special processing
                     03A8   510                                           ;   for F close to 1
                     03A8   511
                     03A8   512 ;
                     03A8   513 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                     03A8   514 ;
                     03A8   515
          7E  58 4DFD  03A8   516          CVTWG   R8, -(SP)                ; Push N onto the stack
    58  FC4F CF43  7E  03AC   517          MOVAQ   MTH$$AB_G_FHI[R3], R8    ; R8 = Address of FHI
        54   68  7D  03B2   518          MOVQ    (R8), R4                 ; R4/R5 = FHI
    56  50   54 43FD  03B5   519          SUBG3   R4, R0, R6               ; R6/R7 = F - FHI
        50   54 40FD  03BA   520          ADDG2   R4, R0                   ; R0/R1 = F + FHI
        56   50 46FD  03BE   521          DIVG2   R0, R6                   ; R6/R7 = Z = (F - FHI)/(F + FHI)
```

```
                50    56    56 45FD   03C2   522          MULG3   R6, R6, R0                      ; R0/R1 = Z**2
          FEAA  CF    05    50 55FD   03C7   523          POLYG   R0, #LOGLEN2-1, LOGTAB2          ; R0/R1 = P(Z**2)
                      50    56 44FD   03CE   524          MULG2   R6, R0                          ; R0/R1 = Z*P(Z**2)
                                      03D2   525
                                      03D2   526 ;
                                      03D2   527 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                                      03D2   528 ;
          52    FED8  CF    6E    45FD   03D2   529          MULG3   (SP), G_LN_2_LO, R2             ; R2/R3 = N*LN_2_LO
                      52    78    40FD   03D9   530          ADDG2   -(R8), R2                       ; R2/R3 = N*LN_2_LO + LN_FHI_LO
                      50    52    40FD   03DD   531          ADDG2   R2, R0                          ; R0/R1 = B
                                      03E1   532
                                      03E1   533 ;
                                      03E1   534 ; Compute A = N*LN_2_HI + LN_FHI_HI and GLOG(X)
                                      03E1   535 ;
          52    FEC1  CF    8E    45FD   03E1   536          MULG3   (SP)+, G_LN_2_HI, R2            ; R2/R3 = N*LN_2_HI
                      52    78    42FD   03E8   537          SUBG2   -(R8), R2                       ; R2/R3 = A = N*LN_2_HI + LN_FHI_HI
                      50    52    40FD   03EC   538          ADDG2   R2, R0                          ; R0/R1 = A + B = GLOG(X)
                                  05    03F0   539          RSB
                                      03F1   540
                                      03F1   541 ;
                                      03F1   542 ; Special logic for F close to 1
                                      03F1   543 ;
                                      03F1   544
                                      03F1   545 LN_1_PLUS_W:
                50    50    08 43FD   03F1   546          SUBG3   #1, R0, R6                      ; R6/R7 = W = F - 1
          FE33  CF    08    56 55FD   03F6   547          POLYG   R6, #LOGLEN1-1,LOGTAB1          ; R0/R1 = Q(W)
                      50    56 44FD   03FD   548          MULG2   R6, R0                          ; R0/R1 = W*Q(W)
                      54    58 4DFD   0401   549          CVTWG   R8, R4                          ; R4/R5 = N
          52    FEA5  CF    54    45FD   0405   550          MULG3   R4, G_LN_2_LO, R2               ; R2/R3 = N*LN_2_LO
                      50    52    40FD   040C   551          ADDG2   R2, R0                          ; R0/R1 = N*LN_2_LO + W*Q(W)
                      50    56    40FD   0410   552          ADDG2   R6, R0                          ; R0/R1 = N*LN_2_LO + LN(F)
                54    FE8F  CF    44FD   0414   553          MULG2   G_LN_2_HI, R4                   ; R4/R5 = N*LN_2_HI
                      50    54    40FD   041A   554          ADDG2   R4, R0                          ; R0/R1 = GLOG(X)
                                  05    041E   555          RSB
                                      041F   556
                                      041F   557 ;
                                      041F   558 ; X =< 0.0, signal error
                                      041F   559 ;
                          6E    DD    041F   560 ERROR:     PUSHL   (SP)                            ; return PC from JSB routine
                  7E      00'8F   9A  0421   561          MOVZBL  #MTH$K_LOGZERNEG, -(SP)         ; condition value
                50    01    0F    79  0425   562          ASHQ    #15, #1, R0                     ; R0 = result = reserved operand -0.0
                                      0429   563                                                  ; goes to signal mechanism vector
                                      0429   564                                                  ; (CHF$L_MCH_R0/R1) so error handler
                                      0429   565                                                  ; can modify the result.
 00000000'GF    02    FB    0429   566          CALLS   #2, G^MTH$$SIGNAL                       ; signal error and use real user's PC
                                      0430   567                                                  ; independent of CALL vs JSB
                                  05  0430   568          RSB                                     ; return - R0 restored from
                                      0431   569                                                  ; CHF$L_MCH_R0/R1
                                      0431   570
                                      0431   571
                                      0431   572
                                      0431   573          .END
```

```
ACMASK          =  000041FC
ERR                0000030E  R      01
ERROR              0000041F  R      01
GLOG_COMMON_R8     00000317  R      01
G_GLOG10_E         000002B8  R      01
G_INV_LN2_CONS     000002C0  R      01
G_LN_2_HI          000002A8  R      01
G_LN_2_LO          000002B0  R      01
LN_1_PLUS          00000388  R      01
LN_1_PLUS_W        000003F1  R      01
LOGLEN1         =  00000009
LOGLEN2         =  00000006
LOGTAB1            00000230  R      01
LOGTAB2            00000278  R      01
LONG            =  00000004
MTH$$AB_ALOG_V     ********    X    00
MTH$$AB_G_FHI      00000000  RG     01
MTH$$JACKET_HND    ********    X    01
MTH$$SIGNAL        ********    X    00
MTH$GLOG           000002C8  RG     01
MTH$GLOG10         000002D9  RG     01
MTH$GLOG10_R8      00000300  RG     01
MTH$GLOG2          000002EA  RG     01
MTH$GLOG_R8        00000311  RG     01
MTH$K_LOGZERNEG    ********    X    00
NEG_EXP            0000038A  R      01
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	00000431 (1073.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.10	00:00:00.73
Command processing	118	00:00:00.71	00:00:03.58
Pass 1	101	00:00:01.62	00:00:06.01
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	112	00:00:01.33	00:00:06.43
Symbol table output	3	00:00:00.04	00:00:00.06
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	367	00:00:03.83	00:00:16.84

The working set limit was 1050 pages.
9346 bytes (19 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 27 non-local and 0 local symbols.
633 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$:MTHGLOG/OBJ=OBJ$:MTHGLOG MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC$:
```

MTHGCONJG
LIS

MTHGINT
LIS

MTHGMOD
LIS

MTHEXP
LIS

MTHFLOOR
LIS

MTHDTAN
LIS

MTHDTANH
LIS

MTHGEXP
LIS

MTHGFLOOR
LIS

MTHGMIN1
LIS

MTHGCOSH
LIS

MTHGLOG
LIS

MTHGACOS
LIS

MTHGASIN
LIS

MTHGNINT
LIS

MTHGATAN
LIS

MTHGATANH
LIS

MTHGMAX1
LIS