```
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    AAAAAA    LL            OOOOOO      GGGGGGGG
MM      MM  TTTTTTTTTT  HH      HH    AAAAAA    LL            OOOOOO      GGGGGGGG
MMMM  MMMM      TT      HH      HH  AA      AA  LL          OO      OO  GG
MMMM  MMMM      TT      HH      HH  AA      AA  LL          OO      OO  GG
MM  MM  MM      TT      HH      HH  AA      AA  LL          OO      OO  GG
MM  MM  MM      TT      HH      HH  AA      AA  LL          OO      OO  GG
MM      MM      TT      HHHHHHHHHH  AA      AA  LL          OO      OO  GG
MM      MM      TT      HHHHHHHHHH  AA      AA  LL          OO      OO  GG
MM      MM      TT      HH      HH  AAAAAAAAAA  LL          OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  AAAAAAAAAA  LL          OO      OO  GG  GGGGGG
MM      MM      TT      HH      HH  AA      AA  LL          OO      OO  GG      GG      ....
MM      MM      TT      HH      HH  AA      AA  LL          OO      OO  GG      GG      ....
MM      MM      TT      HH      HH  AA      AA  LLLLLLLLLL    OOOOOO      GGGGGG        ....
MM      MM      TT      HH      HH  AA      AA  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$ALOG            ; Floating Point Natural and Common
0000     2                                      ; Logarithm Functions (ALOG,ALOG10)
0000     3          .IDENT /2-004/              ; File: MTHALOG.MAR  Edit: PDG2004
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$ALOG and MTH$ALOG10 are functions which return the floating point
0000    34 ; natural  or common logarithm of their single precision floating point
0000    35 ; argument. The call is standard call-by-reference.
0000    36 ; MTH$ALOG_R5  and  MTH$ALOG10_R5   are  special routines which are the
0000    37 ; same as MTH$ALOG and MTH$ALOG10 except a faster non-standard JSB call
0000    38 ; is 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 ;
0000    53 ; VERSION: 02
0000    54 ;
0000    55 ; HISTORY:
0000    56 ; AUTHOR:
0000    57 ;       Bob Hanek, 02-JUN-81: Version 02
```

```
                        58 ;
                        59 ;
```

```
0000     61          .SBTTL  HISTORY   Detailed Current Edit History
0000     62
0000     63 ; VERSION 1
0000     64 ;
0000     65 ;
0000     66 ; ALGORITHMIC DIFFERENCE FROM FP-11C ROUTINE:
0000     67 ;       1. Uses POLYF so greater accuracy.
0000     68 ;
0000     69 ; Edit History for Version 01 of MTH$ALOGLOG10
0000     70 ;
0000     71 ; 01-1  Code saving after code review March 1977
0000     72 ; 01-2  Finished error handling 10 June 1977
0000     73 ; 0-3   MTH$$ERROR changed to MTH$$SIGNAL.
0000     74 ;       MTH$_... changed to MTH__....
0000     75 ;       Changed error handling mechanism. Put error result in R0 before
0000     76 ;       calling MTH$$SIGNAL in order to allow user modify error result.
0000     77 ; 01-6  Move .ENTRY mask definition.  TNH 14-Aug-78
0000     78 ; 1-007 - Update version number and copyright notice.  JBS 16-NOV-78
0000     79 ; 1-008 - Change MTH_LOGZERNEG to MTH$K_LOGZERNEG.  JBS 07-DEC-78
0000     80 ; 1-009 - Add "_" to PSECT directive.  JBS 21-DEC-78
0000     81 ; 1-010 - Add comment explaining code trickery.  SBL 14-Feb-1979
0000     82 ; 1-011 - Declare externals.  SBL 17-May-1979
0000     83 ;
0000     84 ;
0000     85 ; Edit History for Version 02 of MTH$ALOGLOG10
0000     86 ;
0000     87 ; 2-001 - Add log base 2 entry points.  RNH 08-Aug-1981
0000     88 ; 2-002 - Use general mode addressing for externals.  SBL 24-Aug-1981
0000     89 ; 2-003 - Added vectored entry point MTH$$AB_ALOG_V.  RNH 29-Sep-81
0000     90 ; 2-004 - Change F_FHI to the global symbol MTH$$AB_F_FHI.  PDG 3-Nov-81
```

```
                        92          .SBTTL  DECLARATIONS    ; Declarative Part of Module
                        93
                        94 ;
                        95 ; INCLUDE FILES:         MTHJACKET.MAR
                        96 ;
                        97
                        98 ;
                        99 ; EXTERNAL SYMBOLS:
                       100 ;
                       101          .DSABL  GBL
                       102          .EXTRN  MTH$K_LOGZERNEG
                       103          .EXTRN  MTH$$SIGNAL
                       104
                       105 ;
                       106 ; EQUATED SYMBOLS:
                       107
                       108
0000003C               109          ACMASK = ^M<R2, R3, R4, R5>     ; register entry mask and integer
                       110                                          ; overflow enable
72003CB1               111          LF_LN_2_HI = ^X72003CB1         ; (Hi 16 bits of ln2)*2**-7
BE8E333F               112          LF_LN_2_LO = ^XBE8E333F         ; (Low bits of ln2)*2**-7
5BD93FDE               113          LF_LOG10_E = ^X5BD93FDE         ; log10(e)
00004080               114          SF_1       = ^F1.0
                       115
AA3B40B8               116          F_INV_LN2_CONS  = ^F1.44269504088896340735992468100189213742 66
                       117                                          ; convert natural log to log base 2
                       118 ;
                       119 ; MACROS:       none
                       120 ;
                       121 ; PSECT DECLARATIONS:
                       122
                  00000123          .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                       124                                          ; program section for math routines
                       125 ;
                       126 ; OWN STORAGE:  none
                       127 ;
```

```
                                0000   129 ; CONSTANTS:
                                0000   130
                                0000   131 ;
                                0000   132 ;   The MTH$$AB_ALOG table is accessed by the low order exponent  bit  and the
                                0000   133 ;   first 7 fraction bits (not including the hidden bit) of the argument.  The
                                0000   134 ;   table  entries  are  offsets  into  the  F_FHI table below.  Note that the
                                0000   135 ;   MTH$$AB_ALOG table is data type independent and is used by  all  four  LOG
                                0000   136 ;   routines.
                                0000   137
                                0000   138
                                0000   139 MTH$$AB_ALOG_V::                    ; Simulated vector entry for G
                    00000004'   0000   140         .LONG   MTH$$AB_ALOG-.     ;   and H routines
                                0004   141
                                0004   142 MTH$$AB_ALOG::
04 04 04 04 04 04 04 04         0004   143         .BYTE   ^X04, ^X04, ^X04, ^X04, ^X04, ^X04, ^X04, ^X04
04 04 04 04 04 04 04 04         000C   144         .BYTE   ^X04, ^X04, ^X04, ^X04, ^X04, ^X04, ^X04, ^X04
09 09 09 04 04 04 04 04         0014   145         .BYTE   ^X04, ^X04, ^X04, ^X04, ^X04, ^X09, ^X09, ^X09
09 09 09 09 09 09 09 09         001C   146         .BYTE   ^X09, ^X09, ^X09, ^X09, ^X09, ^X09, ^X09, ^X09
09 09 09 09 09 09 09 09         0024   147         .BYTE   ^X09, ^X09, ^X09, ^X09, ^X09, ^X09, ^X09, ^X09
0E 0E 0E 0E 0E 0E 0E 0E         002C   148         .BYTE   ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E
0E 0E 0E 0E 0E 0E 0E 0E         0034   149         .BYTE   ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E, ^X0E
13 13 13 13 13 13 13 13         003C   150         .BYTE   ^X13, ^X13, ^X13, ^X13, ^X13, ^X13, ^X13, ^X13
18 18 13 13 13 13 13 13         0044   151         .BYTE   ^X13, ^X13, ^X13, ^X13, ^X13, ^X13, ^X18, ^X18
18 18 18 18 18 18 18 18         004C   152         .BYTE   ^X18, ^X18, ^X18, ^X18, ^X18, ^X18, ^X18, ^X18
1D 1D 1D 1D 1D 1D 1D 18         0054   153         .BYTE   ^X18, ^X1D, ^X1D, ^X1D, ^X1D, ^X1D, ^X1D, ^X1D
22 22 22 22 22 22 1D 1D         005C   154         .BYTE   ^X1D, ^X1D, ^X22, ^X22, ^X22, ^X22, ^X22, ^X22
2C 27 27 27 27 27 27 22         0064   155         .BYTE   ^X22, ^X27, ^X27, ^X27, ^X27, ^X27, ^X27, ^X2C
31 31 31 31 2C 2C 2C 2C         006C   156         .BYTE   ^X2C, ^X2C, ^X2C, ^X2C, ^X31, ^X31, ^X31, ^X31
45 40 40 3B 3B 36 36 36         0074   157         .BYTE   ^X36, ^X36, ^X36, ^X3B, ^X3B, ^X40, ^X40, ^X45
FF FF FF FF FF FF FF FF         007C   158         .BYTE   ^XFF, ^XFF, ^XFF, ^XFF, ^XFF, ^XFF, ^XFF, ^XFF
32 37 3C 41 FF FF FF FF         0084   159         .BYTE   ^XFF, ^XFF, ^XFF, ^XFF, ^X41, ^X3C, ^X37, ^X32
23 23 28 28 28 2D 2D 32         008C   160         .BYTE   ^X32, ^X2D, ^X2D, ^X28, ^X28, ^X28, ^X23, ^X23
19 1E 1E 1E 1E 1E 23 23         0094   161         .BYTE   ^X23, ^X23, ^X1E, ^X1E, ^X1E, ^X1E, ^X1E, ^X19
14 14 19 19 19 19 19 19         009C   162         .BYTE   ^X19, ^X19, ^X19, ^X19, ^X19, ^X19, ^X14, ^X14
0F 14 14 14 14 14 14 14         00A4   163         .BYTE   ^X14, ^X14, ^X14, ^X14, ^X14, ^X14, ^X14, ^X0F
0F 0F 0F 0F 0F 0F 0F 0F         00AC   164         .BYTE   ^X0F, ^X0F, ^X0F, ^X0F, ^X0F, ^X0F, ^X0F, ^X0F
0A 0A 0A 0A 0A 0F 0F 0F         00B4   165         .BYTE   ^X0F, ^X0F, ^X0F, ^X0A, ^X0A, ^X0A, ^X0A, ^X0A
0A 0A 0A 0A 0A 0A 0A 0A         00BC   166         .BYTE   ^X0A, ^X0A, ^X0A, ^X0A, ^X0A, ^X0A, ^X0A, ^X0A
05 05 05 05 0A 0A 0A 0A         00C4   167         .BYTE   ^X0A, ^X0A, ^X0A, ^X0A, ^X05, ^X05, ^X05, ^X05
05 05 05 05 05 05 05 05         00CC   168         .BYTE   ^X05, ^X05, ^X05, ^X05, ^X05, ^X05, ^X05, ^X05
05 05 05 05 05 05 05 05         00D4   169         .BYTE   ^X05, ^X05, ^X05, ^X05, ^X05, ^X05, ^X05, ^X05
00 00 00 00 05 05 05 05         00DC   170         .BYTE   ^X05, ^X05, ^X05, ^X05, ^X00, ^X00, ^X00, ^X00
00 00 00 00 00 00 00 00         00E4   171         .BYTE   ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00
00 00 00 00 00 00 00 00         00EC   172         .BYTE   ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00
00 00 00 00 00 00 00 00         00F4   173         .BYTE   ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00
00 00 00 00 00 00 00 00         00FC   174         .BYTE   ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00, ^X00
                                0104   175
```

```
                     0104   177 ;
                     0104   178 ;
                     0104   179 ;   The F_FHI table is accessed by an index obtained from the MTH$$AB_ALOG
                     0104   180 ;   table.  Indices between 0 and 13 inclusive are used to access  entries
                     0104   181 ;   0  through 13 respectively.  For these indices, the first three items
                     0104   182 ;   of the corresponding entry are  FHI,  LN_FHI_LO  and  LN_FHI_HI .  The
                     0104   183 ;   last two items for these entries are not used.  Indices between 14 and
                     0104   184 ;   27 inclusive access entries 13 through 0 respectively.  For  these in-
                     0104   185 ;   dices, the last three items in the corresponding entry are LN_FHI_HI,
                     0104   186 ;   LN_FHI_LO and FHI.  The first two items for these entries are not used.
                     0104   187 ;
                     0104   188
                     0104   189 MTH$$AB_F_FHI::
                     0104   190 ; Entry  0
4F9040ED             0104   191          .LONG    ^X4F9040ED      ; .18539906E+01
4C703A20             0108   192          .LONG    ^X4C703A20      ; .15287264E-03
0000401E             010C   193          .LONG    ^X0000401E      ; .61718750E+00
42A4BA20             0110   194          .LONG    ^X42A4BA20      ;-.15283615E-03
149E400A             0114   195          .LONG    ^X149E400A      ; .53937709E+00
                     0118   196 ; Entry  1
1D4340CF             0118   197          .LONG    ^X1D4340CF      ; .16180805E+01
3D7A3925             011C   198          .LONG    ^X3D7A3925      ; .39396320E-04
60003FF6             0120   199          .LONG    ^X60003FF6      ; .48120117E+00
8565B925             0124   200          .LONG    ^X8565B925      ;-.39463299E-04
364F401E             0128   201          .LONG    ^X364F401E      ; .61801618E+00
                     012C   202 ; Entry  2
68D440BA             012C   203          .LONG    ^X68D440BA      ; .14563241E+01
FE6E3A3F             0130   204          .LONG    ^XFE6E3A3F      ; .18309962E-03
60003FC0             0134   205          .LONG    ^X60003FC0      ; .37573242E+00
FA3BBA3F             0138   206          .LONG    ^XFA3BBA3F      ;-.18308398E-03
C8F9402F             013C   207          .LONG    ^XC8F9402F      ; .68666035E+00
                     0140   208 ; Entry  3
AD1D40AB             0140   209          .LONG    ^XAD1D40AB      ; .13412205E+01
F63E3A00             0144   210          .LONG    ^XF63E3A00      ; .12298764E-03
40003F96             0148   211          .LONG    ^X40003F96      ; .29345703E+00
F804BA00             014C   212          .LONG    ^XF804BA00      ;-.12299424E-03
DEF5403E             0150   213          .LONG    ^XDEF5403E      ; .74558955E+00
                     0154   214 ; Entry  4
1DA240A1             0154   215          .LONG    ^X1DA240A1      ; .12587168E+01
96E539EB             0158   216          .LONG    ^X96E539EB      ; .11233780E-03
80003F6B             015C   217          .LONG    ^X80003F6B      ; .22998047E+00
8F32B9EB             0160   218          .LONG    ^X8F32B9EB      ;-.11232345E-03
61B9404B             0164   219          .LONG    ^X61B9404B      ; .79445988E+00
                     0168   220 ; Entry  5
8BD24099             0168   221          .LONG    ^X8BD24099      ; .11995795E+01
F8D539B4             016C   222          .LONG    ^XF8D539B4      ; .86294174E-04
40003F3A             0170   223          .LONG    ^X40003F3A      ; .18188477E+00
E576B9B4             0174   224          .LONG    ^XE576B9B4      ;-.86258093E-04
687B4055             0178   225          .LONG    ^X687B4055      ; .83362550E+00
                     017C   226 ; Entry  6
FFA64093             017C   227          .LONG    ^XFFA64093      ; .11562393E+01
90C63902             0180   228          .LONG    ^X90C63902      ; .31129246E-04
A0003F14             0184   229          .LONG    ^XA0003F14      ; .14514160E+00
5C1CB902             0188   230          .LONG    ^X5C1CB902      ;-.31080199E-04
6850405D             018C   231          .LONG    ^X6850405D      ; .86487293E+00
                     0190   232 ; Entry  7
C18C408F             0190   233          .LONG    ^XC18C408F      ; .11230941E+01
```

```
0DCB397A  0194   234          .LONG   ^X0DCB397A      ; .59617490E-04
A0003EED  0198   235          .LONG   ^XA0003EED      ; .11602783E+00
FBF5B979  019C   236          .LONG   ^XFBF5B979      ;-.59600879E-04
F1154063  01A0   237          .LONG   ^XF1154063      ; .89039737E+00
          01A4   238 ; Entry  8
5B39408C  01A4   239          .LONG   ^X5B39408C      ; .10965339E+01
BAC4395A  01A8   240          .LONG   ^XBAC4395A      ; .52149189E-04
A0003EBC  01AC   241          .LONG   ^XA0003EBC      ; .92102051E-01
E725B95A  01B0   242          .LONG   ^XE725B95A      ;-.52190520E-04
76814069  01B4   243          .LONG   ^X76814069      ; .91196448E+00
          01B8   244 ; Entry  9
B2B24089  01B8   245          .LONG   ^XB2B24089      ; .10757658E+01
DE8A3911  01BC   246          .LONG   ^XDE8A3911      ; .34777950E-04
80003E95  01C0   247          .LONG   ^X80003E95      ; .72998047E-01
98DAB911  01C4   248          .LONG   ^X98DAB911      ;-.34713048E-04
F853406D  01C8   249          .LONG   ^XF853406D      ; .92957038E+00
          01CC   250 ; Entry 10
B4DF4087  01CC   251          .LONG   ^XB4DF4087      ; .10602072E+01
FB5338C2  01D0   252          .LONG   ^XFB5338C2      ; .23243634E-04
60003E6F  01D4   253          .LONG   ^X60003E6F      ; .58441162E-01
9E7BB8C2  01D8   254          .LONG   ^X9E7BB8C2      ;-.23200400E-04
76554071  01DC   255          .LONG   ^X76554071      ; .94321185E+00
          01E0   256 ; Entry 11
54244086  01E0   257          .LONG   ^X54244086      ; .10494428E+01
23F83839  01E4   258          .LONG   ^X23F83839      ; .11035234E-04
A0003E45  01E8   259          .LONG   ^XA0003E45      ; .48248291E-01
495DB83A  01EC   260          .LONG   ^X495DB83A      ;-.11103545E-04
F0604073  01F0   261          .LONG   ^XF0604073      ; .95288658E+00
          01F4   262 ; Entry 12
38494085  01F4   263          .LONG   ^X38494085      ; .10407802E+01
CA5938C1  01F8   264          .LONG   ^XCA5938C1      ; .23101618E-04
A0003E23  01FC   265          .LONG   ^XA0003E23      ; .39947510E-01
A046B8C1  0200   266          .LONG   ^XA046B8C1      ;-.23082026E-04
F8264075  0204   267          .LONG   ^XF8264075      ; .96081769E+00
          0208   268 ; Entry 13
6EE94084  0208   269          .LONG   ^X6EE94084      ; .10346347E+01
E75E38B2  020C   270          .LONG   ^XE75E38B2      ; .21326992E-04
60003E0B  0210   271          .LONG   ^X60003E0B      ; .34027100E-01
B60EB8B2  0214   272          .LONG   ^XB60EB8B2      ;-.21304029E-04
6E2A4077  0218   273          .LONG   ^X6E2A4077      ; .96652472E+00
          021C   274
          021C   275 ;
          021C   276 ;  Polynomial constants tables
          021C   277 ;
          021C   278
          021C   279
          021C   280 LOGTAB1:                       ; Constants for q(z).  Generated using eq.
          021C   281                                ; 6.3.10 of Hart et. al. (sin(2a) = 1/32)
EABDBF2A  021C   282          .LONG   ^XEABDBF2A      ; C5 = -.16691108
0CDD3F4D  0220   283          .LONG   ^X0CDD3F4D      ; C4 = 0.20024438
FFF6BF7F  0224   284          .LONG   ^XFFF6BF7F      ; C3 = -.24999985
AAA73FAA  0228   285          .LONG   ^XAAA73FAA      ; C2 = 0.33333322
0000C000  022C   286          .LONG   ^X0000C000      ; C1 = -.50000000
00000000  0230   287          .LONG   ^X00000000      ; C0 =  .00000000
00000006  0234   288 LOGLEN1 = .-LOGTAB1/4            ; no. of floating point entries
          0234   289
          0234   290
```

```
                     0234   291 LOGTAB2:                            ; Constants for p(z*z).  Generated using
                     0234   292                                     ; eq. 6.3.11 of Hart et. al. (sin(2a) =
                     0234   293                                     ; (b - 1)/(b + 1) where b = 2**(1/7))
6D943FCD             0234   294          .LONG    ^X6D943FCD       ; C2 = 0.40122664
AA91402A             0238   295          .LONG    ^XAA91402A       ; C1 = 0.66666514
00004100             023C   296          .LONG    ^X00004100       ; C0 = 2.00000000
00000003             0240   297 LOGLEN2 = .-LOGTAB2/4
                     0240   298
```

```
                        .SBTTL  MTH$ALOG  - Standard Single Precision Floating LOG

                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;
                ; LOG  - single precision floating point function
                ;
                ; LOG(X) is computed using the following approximation technique:
                ;
                ;       If X =< 0, error.  Otherwise
                ;
                ;       Let X = f * (2**n), where 1/2 <= f < 1
                ;
                ;       If n is greater than or equal to 1 than
                ;            set N = n - 1 and F = 2*f.
                ;          Else
                ;            set N = n and F = f.
                ;
                ;       Then  ln(x) = N*ln2 + ln(F)
                ;
                ;       If |F - 1| < 2**-5 then
                ;            ln(F) =  W + W*P(W), where W = F - 1 and P
                ;            is a polynomial of degree 5.
                ;          Else
                ;            ln(F) = ln(FHI) + Z*Q(Z*Z), where FHI is ob-
                ;            tained by table look-up, Q is a polynomial of
                ;            degree 2 and Z = (F - FHI)/(F + FHI)
                ;
                ;       NOTE: The quantities ln(FHI) and ln2 are used in the above
                ;             equations in two parts - a high part (containing the
                ;             high order bits) and a low part (containing the low
                ;             order bits.  In the code the high and low parts of the
                ;             constants are indicated by a _HI and _LO suffix respec-
                ;             tively.  The values were chosen such that N*LN_2_HI +
                ;             LN_FHI_HI is exactly representable.
                ;
                ; CALLING SEQUENCE:
                ;
                ;       logarithm.wf.v = MTH$ALOG(x.rf.r)
                ;
                ; INPUT PARAMETERS:
                ;
00000004        ;       LONG = 4                        ; define longword multiplier
00000004        ;       x = 1 * LONG                    ; Contents of x is the argument
                ;
                ; IMPLICIT INPUTS:      none
                ;
                ; OUTPUT PARAMETERS:
                ;
                ;       VALUE:  floating logarithm of the argument
                ;
                ; IMPLICIT OUTPUTS:     none
                ;
                ; COMPLETION CODES:     none
                ;
                ; SIDE EFFECTS:
```

```
                                0240   357 ;
                                0240   358 ; Signals: MTH$_LOGZERNEG if !X! =< 0.0 with reserved operand in R0 (copied to
                                0240   359 ; the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                                0240   360 ; Associated message is: "LOGARITHM OF ZERO OR NEGATIVE VALUE". Result is
                                0240   361 ; reserved operand -0.0 unless a user supplied (or any) error handler changes
                                0240   362 ; CHF$L_MCH_R0/R1.
                                0240   363 ;
                                0240   364 ; NOTE: This procedure disables floating point underflow and integer
                                0240   365 ; overflow, causes no floating overflow or other arithmetic traps, and
                                0240   366 ; preserves enables across the call.
                                0240   367 ;
                                0240   368 ;---
                                0240   369
                                0240   370
                         003C   0240   371         .ENTRY  MTH$ALOG, ACMASK          ; standard call-by-reference entry
                                0242   372                                           ; disable DV (and FU), enable IV
                                0242   373         MTH$FLAG_JACKET                   ; flag that this is a jacket procedure
                                0242
    6D  00000000'GF     9E      0242               MOVAB   G^MTH$$JACKET_HND, (FP)
                                0249                                              ; set handler address to jacket
                                0249                                              ; handler
                                0249
                                0249   374                                        ; in case of an error in special JSB
                                0249   375                                        ; routine
        50     04 BC     50     0249   376         MOVF    @x(AP), R0             ; R0 = arg
                  3A     10     024D   377         BSBB    MTH$ALOG_R5            ; call special LOG rountine
                         04     024F   378         RET                            ; return - result in R0
                                0250   379
```

```
                                0250   381          .SBTTL  MTH$ALOG10  - Standard Single Precision Floating Common Log
                                0250   382
                                0250   383 ;++
                                0250   384 ; FUNCTIONAL DESCRIPTION:
                                0250   385 ;
                                0250   386 ; LOG10  - single precision floating point function
                                0250   387 ;
                                0250   388 ; LOG10(X) is computed as LOG10(E) * LOG(X).
                                0250   389 ;
                                0250   390 ; See description of MTH$ALOG
                                0250   391 ;
                                0250   392 ; CALLING SEQUENCE:
                                0250   393 ;
                                0250   394 ;       logarithm_base_10.wf.v = MTH$ALOG10(x.rf.r)
                                0250   395 ;
                                0250   396 ; INPUT PARAMETERS:
                                0250   397
                       00000004 0250   398          LONG = 4                        ; define longword multiplier
                       00000004 0250   399          x = 1 * LONG                    ; Contents of x is the argument
                                0250   400
                                0250   401 ;
                                0250   402 ; SIDE EFFECTS: See description of MTH$ALOG
                                0250   403 ;
                                0250   404 ;--
                                0250   405
                                0250   406
                           003C 0250   407          .ENTRY  MTH$ALOG10, ACMASK      ; standard call-by-reference entry
                                0252   408                                          ; disable DV (and FU), enable IV
                                0252   409          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                                0252
      6D    00000000'GF    9E  0252                MOVAB   G^MTH$$JACKET_HND, (FP)
                                0259                                               ; set handler address to jacket
                                0259                                               ; handler
                                0259
                                0259   410                                         ; in case of an error in special JSB
                                0259   411                                         ; routine
      50    04 BC    50       0259   412          MOVF    @x(AP), R0              ; R0 = arg
                18    10       025D   413          BSBB    MTH$ALOG10_R5           ; call special LOG10 routine
                      04       025F   414          RET                             ; return - result in R0
                               0260   415
```

```
                          0260   417          .SBTTL  MTH$ALOG2  - Standard Single Precision Floating Common Logarithm
                          0260   418
                          0260   419 ;++
                          0260   420 ; FUNCTIONAL DESCRIPTION:
                          0260   421 ;
                          0260   422 ; LOG2  - single precision floating point function
                          0260   423 ;
                          0260   424 ; LOG2(X) is computed as LOG2(E) * LOG(X).
                          0260   425 ;
                          0260   426 ; See description of MTH$ALOG
                          0260   427 ;
                          0260   428 ; CALLING SEQUENCE:
                          0260   429 ;
                          0260   430 ;       logarithm_base_2.wf.v = MTH$ALOG2(x.rf.r)
                          0260   431 ;
                          0260   432 ; INPUT PARAMETERS:
                          0260   433
                 00000004 0260   434          LONG = 4                        ; define longword multiplier
                 00000004 0260   435          x = 1 * LONG                    ; Contents of x is the argument
                          0260   436
                          0260   437 ;
                          0260   438 ; SIDE EFFECTS: See description of MTH$ALOG
                          0260   439 ;
                          0260   440 ;--
                          0260   441
                          0260   442
                     003C 0260   443          .ENTRY  MTH$ALOG2, ACMASK       ; standard call-by-reference entry
                          0262   444                                          ; disable DV (and FU), enable IV
                          0262   445          MTH$FLAG_JACKET                 ; flag that this is a jacket procedure
                          0262
      6D  00000000'GF     9E   0262                   MOVAB   G^MTH$$JACKET_HND, (FP)
                          0269                                                ; set handler address to jacket
                          0269                                                ; handler
                          0269
                          0269   446                                          ; in case of an error in special JSB
                          0269   447                                          ; routine
      50      04 BC       50   0269   448          MOVF    @x(AP), R0          ; R0 = arg
                  1A      10   026D   449          BSBB    MTH$ALOG_R5         ; JSB to the natural log routine
      50  AA3B40B8 8F     44   026F   450          MULF2   #F_INV_LN2_CONS, R0 ; convert to base 2
                          04   0276   451          RET                         ; return - result in R0
                               0277   452
                               0277   453
```

```
                     0277   455          .SBTTL  MTH$ALOGLOG10_R5  - Special LOG/LOG10 routines
                     0277   456
                     0277   457 ; Special LOG/LOG10 - used by the standard, and directly.
                     0277   458 ;
                     0277   459 ; CALLING SEQUENCE:
                     0277   460 ;       save anything needed in R0:R5
                     0277   461 ;       MOVF    ...,R0                  ; input in R0
                     0277   462 ;       JSB     MTH$ALOG10_R5 /MTH$ALOG_R5
                     0277   463 ;       return with result in R0
                     0277   464 ; Note: This routine is written to avoid causing any integer overflows,
                     0277   465 ; floating overflows, or floating underflows or divide by 0 conditions,
                     0277   466 ; whether enabled or not.
                     0277   467 ;
                     0277   468 ; REGISTERS USED:
                     0277   469 ;       R0 - Floating argument then result
                     0277   470 ;       R1 - scratch
                     0277   471 ;       R0:R3 - POLYF
                     0277   472 ;       R4 - W or Z during POLYF
                     0277   473 ;       R5 - pointer into F_FHI table
                     0277   474 ;-
                     0277   475
                     0277   476 MTH$ALOG10_R5::                          ; special LOG10 routine
55   50   007F 8F    AB     0277   477          BICW3   #^X7F, R0, R5       ; R5 = Biased exponent
                  6E 15     027D   478          BLEQ    ERR                 ; ALOG10(X) is not defined for X=<0
                            027F   479                                      ; NOTE: User's PC is on top of the
                            027F   480                                      ; stack.  The ERROR logic depends on
                            027F   481                                      ; the user's PC being on top of the
                            027F   482                                      ; stack so a direct call to MTH$ALOG
                            027F   483                                      ; is not used
                10  10      027F   484          BSBB    ALOG_COMMON_R5      ; Call common LOG/LOG10 logic
50   5BD93FDE 8F    44     0281   485          MULF    #LF_LOG10_E, R0     ; R0 = LOG10(e)*LOG(X) = LOG10(X)
                  05        0288   486          RSB                         ; Return
                            0289   487
                            0289   488
                            0289   489 MTH$ALOG_R5::                         ; special LOG routine
55   50   007F 8F    AB     0289   490          BICW3   #^X7F, R0, R5       ; R5 = Biased exponent
                  5C 15     028F   491          BLEQ    ERR                 ; ALOG(X) is not defined for X=<0
                            0291   492 ALOG_COMMON_R5:
     55   4000 8F    A2     0291   493          SUBW    #^X4000, R5         ; R5 = Unbiased exponent
                  58 15     0296   494          BLEQ    NEG_EXP             ; Branch to processing for n=<0
                            0298   495
                            0298   496 ;
                            0298   497 ; Exponent is positive.  N = n - 1 and F = 2f
                            0298   498 ;
                            0298   499
     55   0080 8F    A2     0298   500          SUBW    #^X80, R5           ; R5 = N = n - 1
          50   55    A2     029D   501          SUBW    R5, R0              ; R0 = F = 2f
          53   50    9A     02A0   502          MOVZBL  R0, R3              ; R3 = index into MTH$$AB_ALOG table
53   FD5C CF43    90     02A3   503          MOVB    MTH$$AB_ALOG[R3], R3    ; R3 = offset into F_FHI tables
                  40 19     02A9   504          BLSS    LN_1_PLUS           ; Branch to special processing
                            02AB   505                                      ;   for F close to 1
                            02AB   506
                            02AB   507 ;
                            02AB   508 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                            02AB   509 ;
                            02AB   510
     7E   55    4D     02AB   511          CVTWF   R5, -(SP)               ; Push N onto the stack
```

```
                55    FE51 CF43    DE   02AE   512          MOVAL   MTH$$AB_F_FHI[R3], R5   ; R5 = Address of FHI
                        52   85    D0   02B4   513          MOVL    (R5)+, R2               ; R2 = FHI
                  54    50   52    43   02B7   514          SUBF3   R2, R0, R4              ; R4 = F - FHI
                        50   52    40   02BB   515          ADDF    R2, R0                  ; R0 = F + FHI
                        54   50    46   02BE   516          DIVF    R0, R4                  ; R4 = Z = (F - FHI)/(F + FHI)
                50      54   54    45   02C1   517          MULF3   R4, R4, R0              ; R0 = Z**2
            FF69 CF     02   50    55   02C5   518          POLYF   R0, #LOGLEN2-1, LOGTAB2 ; R0 = P(Z**2)
                        50   54    44   02CB   519          MULF    R4, R0                  ; R0 = Z*P(Z**2)
                                        02CE   520
                                        02CE   521 ;
                                        02CE   522 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                                        02CE   523 ;
          51    BE8E333F 8F    6E    45   02CE   524          MULF3   (SP), #LF_LN_2_LO, R1   ; R1 = N*LN_2_LO
                        51   85    40   02D6   525          ADDF    (R5)+, R1               ; R1 = N*LN_2_LO + LN_FHI_LO
                        50   51    40   02D9   526          ADDF    R1, R0                  ; R0 = B
                                        02DC   527
                                        02DC   528 ;
                                        02DC   529 ; Compute A = N*LN_2_HI + LN_FHI_HI and ALOG(X)
                                        02DC   530 ;
          51    72003CB1 8F    8E    45   02DC   531          MULF3   (SP)+, #LF_LN_2_HI, R1  ; R1 = N*LN_2_HI
                        51   65    40   02E4   532          ADDF    (R5), R1                ; R1 = A = N*LN_2_HI + LN_FHI_HI
                        50   51    40   02E7   533          ADDF    R1, R0                  ; R0 = A + B = ALOG(X)
                                    05   02EA   534          RSB
                                        02EB   535
                                        02EB   536 LN_1_PLUS:
                             51     11   02EB   537          BRB     LN_1_PLUS_W
                                        02ED   538
                           0077     31   02ED   539 ERR:     BRW     ERROR
                                        02F0   540
                                        02F0   541 ;
                                        02F0   542 ; Exponent is negative.  N = n and F = f
                                        02F0   543 ;
                                        02F0   544
                      50    55    A2   02F0   545 NEG_EXP:SUBW    R5, R0                  ; R0 = F = f
                      53    50    9A   02F3   546          MOVZBL  R0, R3                  ; R3 = index into MTH$$AB_ALOG table
                53    FD09 CF43    90   02F6   547          MOVB    MTH$$AB_ALOG[R3], R3    ; R3 = offset into F_FHI tables
                            40     19   02FC   548          BLSS    LN_1_PLOS_W             ; Branch to special processing
                                        02FE   549                                          ;   for F close to 1
                                        02FE   550
                                        02FE   551 ;
                                        02FE   552 ; Compute Z, Z**2, P(Z**2) and Z*P(Z**2)
                                        02FE   553 ;
                                        02FE   554
                      7E    55    4D   02FE   555          CVTWF   R5, -(SP)               ; Push N onto the stack
                55    FDFE CF43    DE   0301   556          MOVAL   MTH$$AB_F_FHI[R3], R5   ; R5 = Address of FHI
                        52   65    D0   0307   557          MOVL    (R5), R2                ; R2 = FHI
                  54    50   52    43   030A   558          SUBF3   R2, R0, R4              ; R4 = F - FHI
                        50   52    40   030E   559          ADDF    R2, R0                  ; R0 = F + FHI
                        54   50    46   0311   560          DIVF    R0, R4                  ; R4 = Z = (F - FHI)/(F + FHI)
                50      54   54    45   0314   561          MULF3   R4, R4, R0              ; R0 = Z**2
            FF16 CF     02   50    55   0318   562          POLYF   R0, #LOGLEN2-1, LOGTAB2 ; R0 = P(Z**2)
                        50   54    44   031E   563          MULF    R4, R0                  ; R0 = Z*P(Z**2)
                                        C321   564
                                        0321   565 ;
                                        0321   566 ; Compute B = N*LN_2_LO + LN_FHI_LO + Z*P(Z*Z)
                                        0321   567 ;
          51    BE8E333F 8F    6E    45   0321   568          MULF3   (SP), #LF_LN_2_LO, R1   ; R1 = N*LN_2_LO
```

```
                     51   75   40  0329   569          ADDF    -(R5), R1                ; R1 = N*LN_2_LO + LN_FHI_LO
                     50   51   40  032C   570          ADDF    R1, R0                   ; R0 = B
                                 032F   571
                                 032F   572 ;
                                 032F   573 ; Compute A = N*LN_2_HI + LN_FHI_HI and ALOG(X)
                                 032F   574 ;
          51 72003CB1 8F   8E    45  032F   575          MULF3   (SP)+, #LF_LN_2_HI, R1   ; R1 = N*LN_2_HI
                     51   75   42  0337   576          SUBF    -(R5), R1                ; R1 = A = N*LN_2_HI + LN_FHI_HI
                     50   51   40  033A   577          ADDF    R1, R0                   ; R0 = A + B = ALOG(X)
                                 05  033D   578          RSB
                                 033E   579
                                 033E   580 ;
                                 033E   581 ; Special logic for F close to 1
                                 033E   582 ;
                                 033E   583
                                 033E   584 LN_1_PLUS_W:
               54    50   08    43  033E   585          SUBF3   #SF_1, R0, R4            ; R4 = W = F - 1
          FED4 CF    05   54    55  0342   586          POLYF   R4, #LOGLEN1-1,LOGTAB1   ; R0 = Q(W)
                     50   54    44  0348   587          MULF    R4, R0                   ; R0 = W*Q(W)
                     55   55    4D  034B   588          CVTWF   R5, R5                   ; R5 = N
          51 BE8E333F 8F   55    45  034E   589          MULF3   R5, #LF_LN_2_LO, R1      ; R1 = N*LN_2_LO
                     50   51    40  0356   590          ADDF    R1, R0                   ; R0 = N*LN_2_LO + W*Q(W)
                     50   54    40  0359   591          ADDF    R4, R0                   ; R0 = N*LN_2_LO + LN(F)
               55 72003CB1 8F    44  035C   592          MULF    #LF_LN_2_HI, R5          ; R5 = N*LN_2_HI
                     50   55    40  0363   593          ADDF    R5, R0                   ; R0 = ALOG(X)
                                 05  0366   594          RSB
                                 0367   595
                                 0367   596
                                 0367   597 ;
                                 0367   598 ; X =< 0.0, signal error
                                 0367   599 ;
                                 0367   600
                        6E   DD  0367   601 ERROR:  PUSHL   (SP)                     ; return PC from JSB routine
                  7E    00'8F   9A  0369   602          MOVZBL  #MTH$K_LOGZERNEG, -(SP)  ; condition value
               50   01    0F   78  036D   603          ASHL    #15, #T, R0              ; R0 = result = reserved operand -0.0
                                 0371   604                                           ; goes to signal mechanism vector
                                 0371   605                                           ; (CHF$L_MCH_R0/R1) so error handler
                                 0371   606                                           ; can modify the result.
00000000'GF      02   FB  0371   607          CALLS   #2, G^MTH$$SIGNAL        ; signal error and use real user's PC
                                 0378   608                                           ; independent of CALL vs JSB
                                 05  0378   609          RSB                      ; return - R0 restored from
                                 0379   610                                           ; CHF$L_MCH_R0/R1
                                 0379   611
                                 0379   612
                                 0379   613          .END
```

```
ACMASK          = 0000003C
ALOG_COMMON_R5    00000291 R      01
ERR               000002ED R      01
ERROR             00000367 R      01
F_INV_LN2_CONS  = AA3B40B8
LF_LN_2_HI      = 72003CB1
LF_LN_2_LO      = BE8E333F
LF_LOG10_E      = 5BD93FDE
LN_1_PLUS         000002EB R      01
LN_1_PLUS_W       0000033E R      01
LOGLEN1         = 00000006
LOGLEN2         = 00000003
LOGTAB1           0000021C R      01
LOGTAB2           00000234 R      01
LONG            = 00000004
MTH$$AB_ALOG      00000004 RG     01
MTH$$AB_ALOG_V    00000000 RG     01
MTH$$AB_F_FHI     00000104 RG     01
MTH$$JACKET_HND   ******** X      01
MTH$$SIGNAL       ******** X      00
MTH$ALOG          00000240 RG     01
MTH$ALOG10        00000250 RG     01
MTH$ALOG10_R5     00000277 RG     01
MTH$ALOG2         00000260 RG     01
MTH$ALOG_R5       00000289 RG     01
MTH$K_LOGZERNEG   ******** X      00
NEG_EXP           000002F0 R      01
SF_1            = 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	00000379 (889.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	36	00:00:00.07	00:00:01.92
Command processing	129	00:00:00.58	00:00:03.81
Pass 1	100	00:00:01.90	00:00:05.88
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	117	00:00:01.30	00:00:03.52
Symbol table output	4	00:00:00.05	00:00:00.05
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	390	00:00:03.93	00:00:15.21

The working set limit was 1050 pages.
10465 bytes (21 pages) of virtual memory were used to buffer the intermediate code.

There were 10 pages of symbol table space allocated to hold 29 non-local and 0 local symbols.
673 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$:MTHALOG/OBJ=OBJ$:MTHALOG MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC$:
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MTH4OVPI
LIS

MTHABS
LIS

MTHAINT
LIS

MTHAMOD
LIS

MTHERR
SDL

MTHASIN
LIS

MTHCDABS
LIS

MTHATAN
LIS

MTHATANH
LIS

MTHCDLOG
LIS

MTHJACKET
MAR

MTHBITOPS
LIS

MTHALOG
LIS

MTHDEF
FOR

MTHANINT
LIS

MTHCABS
LIS

MTHACOS
LIS

MTHCDEXP
LIS