```
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  HH      HH  EEEEEEEEEE  XX      XX  PPPPPPPP
MM       MM  TTTTTTTTTT  HH      HH  HH      HH  EEEEEEEEEE  XX      XX  PPPPPPPP
MMMM   MMMM      TT      HH      HH  HH      HH  EE          XX      XX  PP      PP
MMMM   MMMM      TT      HH      HH  HH      HH  EE          XX      XX  PP      PP
MM  MM  MM       TT      HH      HH  HH      HH  EE            XX  XX    PP      PP
MM  MM  MM       TT      HH      HH  HH      HH  EE            XX  XX    PP      PP
MM      MM       TT      HHHHHHHHHH  HHHHHHHHHH  EEEEEEEE        XX      PPPPPPPP
MM      MM       TT      HHHHHHHHHH  HHHHHHHHHH  EEEEEEEE        XX      PPPPPPPP
MM      MM       TT      HH      HH  HH      HH  EE            XX  XX    PP
MM      MM       TT      HH      HH  HH      HH  EE            XX  XX    PP
MM      MM       TT      HH      HH  HH      HH  EE          XX      XX  PP
MM      MM       TT      HH      HH  HH      HH  EE          XX      XX  PP
MM      MM       TT      HH      HH  HH      HH  EEEEEEEEEE  XX      XX  PP
MM      MM       TT      HH      HH  HH      HH  EEEEEEEEEE  XX      XX  PP

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

MTH
Sym

ACM
APP
EVA
EXC
EXP
HXP
HXP
HXP
HXP
H_1
H_L
H_L
LON
MTH
MTH
MTH
MTH
MTH
MTH
MTH
OVE
SFS
SMT
TAB
TAB
UND
X
X_1

PSE

$AB
_M1

Pha

Int
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
925

Table of contents

```
0000     1         .TITLE  MTH$HEXP        ; H Floating Exponential Function
0000     2                                 ; (HEXP)
0000     3         .IDENT /1-002/          ; File: MTHHEXP.MAR     Edit:RNH1002
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$HEXP is a function which returns the H floating point
0000    34 ; exponential of its H floating point argument.  The call is standard
0000    35 ; call-by-reference.
0000    36 ;
0000    37 ;--
0000    38 ;
0000    39 ; VERSION: 1
0000    40 ;
0000    41 ; HISTORY:
0000    42 ; AUTHOR:
0000    43 ;       John A. Wheeler, 08-Aug-79: Version 1
0000    44 ;
0000    45 ; MODIFIED BY:
0000    46 ;
0000    47 ;
0000    48 ;
```

```
0000     50          .SBTTL  HISTORY ; Detailed Current Edit History
0000     51
0000     52 ; Edit History for Version 1 of MTH$HEXP
0000     53 ;
0000     54 ; 1-001 - Adapted from MTH$GEXP version 1-002.  JAW 08-Aug-79
0000     55 ; 1-002 - Added large argument logic to prevent lose of significance
0000     56 ;         for arguments with absolute value greater than 2**7.
0000     57 ;         RNH 24-JUN-81
```

```
                                59          .SBTTL  DECLARATIONS            ; Declarative Part of Module
                                60
                                61 ;
                                62 ; INCLUDE FILES:        MTHJACKET.MAR
                                63 ;
                                64 ; EXTERNAL SYMBOLS:
                                65 ;
                                66          .DSABL  GBL                     ; Declare all EXTRNs explicitly
                                67          .EXTRN  MTH$$SIGNAL             ; SIGNAL SEVERE error
                                68          .EXTRN  MTH$$JACKET_TST         ; Test to see if called with CALL or
                                69                                          ; JSB
                                70          .EXTRN  MTH$K_FLOUNDMAT         ; Underflow error code
                                71          .EXTRN  MTH$K_FLOOVEMAT         ; Overflow error code
                                72
                                73 ; EQUATED SYMBOLS:
                                74
0000407C                        75          ACMASK  = ^M<IV, R2, R3, R4, R5, R6>    ; .ENTRY mask + int ovf enable
00007D12                        76          X_16LOG2E =     ^X7D12          ; Extension for operand in EMODH
                                77 ;
                                78 ; MACROS:
                                79 ;
                                80          $SFDEF                          ; Define SF$ (stack frame) symbols
                                81 ;
                                82 ; PSECT DECLARATIONS:
                                83
00000000                        84          .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
                                85                                          ; Program section for math routines
                                86 ;
                                87 ; OWN STORAGE:  none
                                88 ;
                                89 ; CONSTANTS:
                                90
                                91 ;
                                92 ; Table of 2**(I/16) for I = 0 to 15
                                93 ;
                                94
00000000 00004001               95 TABHI:   .LONG   ^X00004001,^X00000000   ; 2**(0/16) =
00000000 00000000               96          .LONG   ^X00000000,^X00000000   ;  0.10000000000000000000000000000000E+01
989086CF 0B554001               97          .LONG   ^X0B554001,^X989086CF   ; 2**(1/16) =
42AAB718 8B92F629               98          .LONG   ^X8B92F629,^X42AAB718   ;  0.10442737824274138403219664787399 29E+01
D51783C7 172B4001               99          .LONG   ^X172B4001,^XD51783C7   ; 2**(2/16) =
B14AC50E F7C8ADCD              100          .LONG   ^XF7C8ADCD,^XB14AC50E   ;  0.10905077326652576592070106557607 07E+01
5623A6E7 23874001              101          .LONG   ^X23874001,^X5623A6E7   ; 2**(3/16) =
5CB6B1C1 1FAD866C              102          .LONG   ^X1FAD866C,^X5CB6B1C1   ;  0.11387886347566916537038302838415 11E+01
1B71E0A3 306F4001              103          .LONG   ^X306F4001,^X1B71E0A3   ; 2**(4/16) =
5C864630 8D5A52DE              104          .LONG   ^X8D5A52DE,^X5C864630   ;  0.11892071150027210667174999705604 75E+01
234264C1 3DEA4001              105          .LONG   ^X3DEA4001,^X234264C1   ; 2**(5/16) =
D7743E13 4122235B              106          .LONG   ^X4122235B,^XD7743E13   ;  0.12418578120734840485936774687265 95E+01
62A2AD53 4BFD4001              107          .LONG   ^X4BFD4001,^X62A2AD53   ; 2**(6/16) =
2E21FEC4 397A71D4              108          .LONG   ^X397A71D4,^X2E21FEC4   ;  0.12968395546510096659337541177924 51E+01
85427DD4 5AB040C1              109          .LONG   ^X5AB04001,^X85427DD4   ; 2**(7/16) =
EB345191 9301958C              110          .LONG   ^X9301958C,^XEB345191   ;  0.13542555469368927282980147401407 02E+01
F3BCE667 6A094001              111          .LONG   ^X6A094001,^XF3BCE667   ; 2**(8/16) =
EA951366 B2FBC908              112          .LONG   ^XB2FBC908,^XEA951366   ;  0.14142135623730950488016887242096 97E+01
B018473E 7A114001              113          .LONG   ^X7A114001,^XB018473E   ; 2**(9/16) =
DA1F3F6C 51026D7D              114          .LONG   ^X51026D7D,^XDA1F3F6C   ;  0.14768261459394993113869074803740 49E+01
AA0D5422 8ACE4001              115          .LONG   ^X8ACE4001,^XAA0D5422   ; 2**(10/16) =
```

```
C9BBA192 7C55B5BA  00A8   116          .LONG   ^X7C55B5BA,^XC9BBA192   ;  0.154221082540794082361229186209073 4E+01
3F09182A 9C494001  00B0   117          .LONG   ^X9C494001,^X3F09182A   ; 2**(11/16) =
2BE6071F C46B01C7  00B8   118          .LONG   ^XC46B01C7,^X2BE6071F   ;  0.161049033194925430817952066735740 0E+01
AD3AF995 AE894001  00C0   119          .LONG   ^XAE894001,^XAD3AF995   ; 2**(12/16) =
205A1773 734DD5E8  00C8   120          .LONG   ^X734DD5E8,^X205A1773   ;  0.168179283050742908606225095246642 9E+01
5529BDD8 C1994001  00D0   121          .LONG   ^XC1994001,^X5529BDD8   ; 2**(13/16) =
1BA62A09 0CB1C222  00D8   122          .LONG   ^X0CB1C222,^X1BA62A09   ;  0.175625216037329948311216061937531 3E+01
BA488DCF D5814001  00E0   123          .LONG   ^XD5814001,^XBA488DCF   ; 2**(14/16) =
E0DDEB66 A05A725D  00E8   124          .LONG   ^XA05A725D,^XE0DDEB66   ;  0.183400808640934246348708318958828 8E+01
490DFA2A EA4A4001  00F0   125          .LONG   ^XEA4A4001,^X490DFA2A   ; 2**(15/16) =
DB3018F5 F73A9858  00F8   126          .LONG   ^XF73A9858,^XDB3018F5   ;  0.191520656139714729387261127029583 0E+01
                   0100   127
                   0100   128 ;+
                   0100   129 ;       Each entry in TABLO gives the difference between the corresponding
                   0100   130 ;  entry in TABHI and the power of 2 which it approximates, i.e.,
                   0100   131 ;  TABHI[n] + TABLO[n] = 2**(n/16), to 226 bit accuracy.
                   0100   132 ;-
                   0100   133
00000000 00000000  0100   134 TABLO:  .LONG   ^X00000000,^X00000000   ;  0.000000000000000000000000000000000E+00
00000000 00000000  0108   135          .LONG   ^X00000000,^X00000000
DD3058AF F26FBF8F  0110   136          .LONG   ^XF26FBF8F,^XDD3058AF   ; -9.374520292280427421957567419731049E-35
64EB15EA 7575C53D  0118   137          .LONG   ^X7575C53D,^X64EB15EA
7FDFD542 E4803F8F  0120   138          .LONG   ^XE4803F8F,^X7FDFD542   ;  9.112493410125022978511686101672356E-35
AF748F09 A4FC5D61  0128   139          .LONG   ^XA4FC5D61,^XAF748F09
65BF35EA B13FB 3F  0130   140          .LONG   ^XB13FBF8F,^X65BF35EA   ; -8.148468844525851137325691767487803E-35
1A8576FE 86C84825  0138   141          .LONG   ^X86C84825,^X1A8576FE
BC9D3D8C 2134BF8D  0140   142          .LONG   ^X2134BF8D,^XBC9D3D8C   ; -1.359830974688816973749875638246305E-35
46820E93 D0E560AF  0148   143          .LONG   ^XD0E560AF,^X46820E93
F6EF1F51 174DBF8B  0150   144          .LONG   ^X174DBF8B,^XF6EF1F51   ; -3.283170523176998601615065965334027E-36
00C20344 AEEF660E  0158   145          .LONG   ^XAEEF660E,^X00C20344
EA6345D1 FC9CBF8D  0160   146          .LONG   ^XFC9CBF8D,^XEA6345D1   ; -2.391474797689109171622834301602562E-35
CEB04EC6 7D1BA860  0168   147          .LONG   ^X7D1BA860,^XCEB04EC6
4A035F20 76233F8F  0170   148          .LONG   ^X76233F8F,^X4A035F20   ;  7.036756889073265042421737190671412E-35
8A0C8B2F EF6D81A7  0178   149          .LONG   ^XEF6D81A7,^X8A0C8B2F
05D4EB7B F4F83F8F  0180   150          .LONG   ^XF4F83F8F,^X05D4EB7B   ;  9.422242548621832065692116736394105E-35
642C68BD 426749DD  0188   151          .LONG   ^X426749DD,^X642C68BD
5DE5AD9A 7BD03F8F  0190   152          .LONG   ^X7BD03F8F,^X5DE5AD9A   ;  7.143528991563300614523273615092530E-35
5F016CD2 7A08814C  0198   153          .LONG   ^X7A08814C,^X5F016CD2
3999B0F9 F3763F8E  01A0   154          .LONG   ^XF3763F8E,^X3999B0F9   ;  4.696933478358115495309739213201925E-35
006E7686 A36C8251  01A8   155          .LONG   ^XA36C8251,^X006E7686
F7A386BC C894BF8F  01B0   156          .LONG   ^XC894BF8F,^XF7A386BC   ; -8.587318774298247068868655935103793E-35
33A8A216 BD2E9E4D  01B8   157          .LONG   ^XBD2E9E4D,^X33A8A216
D7A9EB99 FEF03F8F  01C0   158          .LONG   ^XFEF03F8F,^XD7A9EB99   ;  9.609733932128012784507558697141504E-35
D6D0AE09 58B5102C  01C8   159          .LONG   ^X58B5102C,^XD6D0AE09
28D91259 9E513F8F  01D0   160            ONG   ^X9E513F8F,^X28D91259   ;  7.792430785695864249456461125169745E-35
E9F9E5CC 00109849  01D8   161          .LONG   ^X00109849,^XE9F9E5CC
98FDD82A 5829BF8F  01E0   162          .LONG   ^X5829BF8F,^X98FDD82A   ; -6.472995147913347230035214575612342E-35
70BA4139 371FE240  01E8   163          .LONG   ^X371FE240,^X70BA4139
610CEA20 F86D3F8D  01F0   164          .LONG   ^XF86D3F8D,^X610CEA20   ;  2.371815422825174835691651228302996E-35
6527BB00 3EB7EEE1  01F8   165          .LONG   ^X3EB7EEE1,^X6527BB00
                   0200   166
                   0200   167
                   0200   168 ;
                   0200   169 ; Constants used in evaluation of polynomials - small arguments
                   0200   170 ;
                   0200   171
DEC1F87C AE853FD8  0200   172 HXPTB1: .LONG   ^XAE853FD8,^XDEC1F87C   ; 7.647630491059372752329451293833116E-13
```

```
4B9443B8 DC5D98EF  0208   173          .LONG   ^XDC5D98EF,^X4B9443B8
75A10496 939E3FDC  0210   174          .LONG   ^X939E3FDC,^X75A10496   ; 1.14714924138536705330434357106l069E-11
25CDBA3D 9AF19596  0218   175          .LONG   ^X9AF19596,^X25CDBA3D
C138612E 61243FE0  0220   176          .LONG   ^X61243FE0,^XC138612E   ; 1.60590438049139392226423274526859OE-10
140FFFAF 304B573E  0228   177          .LONG   ^X304B573E,^X140EFFAF
9F318EF4 1EED3FE4  0230   178          .LONG   ^X1EED3FE4,^X9F318EF4   ; 2.08767569404622963805653075208744OE-9
CD5A1B98 3F0F3F4B  0238   179          .LONG   ^X3F0F3F4B,^XCD5A1B98
5464567F AE643FE7  0240   180          .LONG   ^XAE643FE7,^X5464567F   ; 2.50521083854428761460497712583349 2E-8
D8953C87 5A7615B2  0248   181          .LONG   ^X5A7615B2,^XD8953C87
8A08FB77 27E43FEB  0250   182          .LONG   ^X27E43FEB,^X8A08FB77   ; 2.75573192239874575905032035036362E-7
B6CF00D8 1DC546DE  0258   183          .LONG   ^X1DC546DE,^XB6CF00D8
6C733A55 71DE3FEE  0260   184          .LONG   ^X71DE3FEE,^X6C733A55   ; 2.75573192239858906286482192750817 5E-6
EDA14A28 8C8138E3  0268   185          .LONG   ^X8C8138E3,^XEDA14A28
1A0101A0 A01A3FF1  0270   186          .LONG   ^XA01A3FF1,^X1A0101A0   ; 2.48015873015873015586105849556798 7E-5
C141B47C 52309FF7  0278   187          .LONG   ^X52309FF7,^XC141B47C
1A0101A0 A01A3FF4  0280   188          .LONG   ^XA01A3FF4,^X1A0101A0   ; 1.98412698412698412698415500265713 6E-4
339AD98C 01BBA01A  0288   189          .LONG   ^X01BBA01A,^X339AD98C
16C1C16C 6C163FF7  0290   190          .LONG   ^X6C163FF7,^X16C1C16C   ; 1.38888888888888888888891741693221 5E-3
431467A7 C18F6C16  0298   191          .LONG   ^XC18F6C16,^X431467A7
11111111 11113FFA  02A0   192          .LONG   ^X11113FFA,^X11111111   ; 8.33333333333333333333333331592122E-3
68A010FF 11111111  02A8   193          .LONG   ^X11111111,^X68A010FF
55555555 55553FFC  02B0   194          .LONG   ^X55553FFC,^X55555555   ; 4.16666666666666666666666665273693E-2
046C5532 55555555  02B8   195          .LONG   ^X55555555,^X046C5532
55555555 55553FFE  02C0   196          .LONG   ^X55553FFE,^X55555555   ; 1.66666666666666666666666666666671E-1
556A5555 55555555  02C8   197          .LONG   ^X55555555,^X556A5555
00000000 00004000  02D0   198          .LONG   ^X00004000,^X00000000   ; 5.00000000000000000000000000000025E-1
001B0000 00000000  02D8   199          .LONG   ^X00000000,^X001B0000
00000000 00004001  02E0   200          .LONG   ^X00004001,^X00000000   ; 9.99999999999999999999999999999999E-1
00000000 00000000  02E8   201          .LONG   ^X00000000,^X00000000
00000000 00004001  02F0   202          .LONG   ^X00004001,^X00000000   ; 9.99999999999999999999999999999999E-1
00000000 00000000  02F8   203          .LONG   ^X00000000,^X00000000
         00000010  0300   204 HXPLN1=<.-HXPTB1>/16                      ; No. of entries in table
                   0300   205
                   0300   206 ;
                   0300   207 ; Constants used in evaluation of polynomial - regular args
                   0300   208 ;
                   0300   209
                   0300   210
16B5D6A9 314B3F9D  0300   211 HXPTAB: .LONG   ^X314B3F9D,^X16B5D6A9   ; 9.40766685810735547974487954283736 7E-31
F4E26068 8EE63FD3  0308   212          .LONG   ^X8EE63FD3,^XF4E26068
0891E189 81643FA5  0310   213          .LONG   ^X81643FA5,^X0891E189   ; 3.04022312083458625188167985073212 7E-28
BB46C345 1B98895D  0318   214          .LONG   ^X1B98895D,^XBB46C345
C99F650B C3BD3FAD  0320   215          .LONG   ^XC3BD3FAD,^XC99F650B   ; 9.12281307564508424034763853634080 6E-26
AEA0B58F 73EA4FC9  0328   216          .LONG   ^X73EA4FC9,^XAEA0B58F
2604C730 E8CA3FB5  0330   217          .LONG   ^XE8CA3FB5,^X2604C730   ; 2.52699593896873207085996257855815 1E-23
913CE0B5 39FFB102  0338   218          .LONG   ^X39FFB102,^X913CE0B5
B8EF5158 E4CF3FBD  0340   219          .LONG   ^XE4CF3FBD,^XB8EF5158   ; 6.41640474059492391242500600785501 7E-21
D69EAB10 5C67FBF4  0348   220          .LONG   ^X5C67FBF4,^XD69EAB10
5E803D39 B5253FC5  0350   221          .LONG   ^XB5253FC5,^X5E803D39   ; 1.48110644793483951876552890349010 2E-18
D71F0E10 EA4B0789  0358   222          .LONG   ^XEA4B0789,^XD71F0E10
5C82223A 62C03FCD  0360   223          .LONG   ^X62C03FCD,^X5C82223A   ; 3.07697029554851108407349386021444 5E-16
41D8C305 6C873FD7  0368   224          .LONG   ^X6C873FD7,^X41D8C305
8B0CFC58 FFCB3FD4  0370   225          .LONG   ^XFFCB3FD4,^X8B0CFC58   ; 5.68208612652862082374967126768966 9E-14
0F6A6EB0 FB8F6867  0378   226          .LONG   ^XFB8F6867,^X0F6A6EB0
6C7812F8 43093FDC  0380   227          .LONG   ^X43093FDC,^X6C7812F8   ; 9.18121957384443876411742295767236 0E-12
11C865C5 B0A976F4  0388   228          .LONG   ^XB0A976F4,^X11C865C5
A673FE78 5D873FE3  0390   229          .LONG   ^X5D873FE3,^XA673FE78   ; 1.27158719505581316217539143608524 8E-9
```

```
                 1CE5E79C 17F71107  0398    230          .LONG   ^X17F71107,^X1CE5E79C
                 A4E7B6FB 3B2A3FEA  03A0    231          .LONG   ^X3B2A3FEA,^XA4E7B6FB   ; 1.46761003229194292632737298181447OE-7
                 87FF0B53 CBBE729C  03A8    232          .LONG   ^XCBBE729C,^X87FF0B53
                 4A0B8D70 C6B03FF0  03B0    233          .LONG   ^XC6B03FF0,^X4A0B8D70   ; 1.35508077794974560432476229902933OE-5
                 34F12BB2 3A76F8B3  03B8    234          .LONG   ^X3A76F8B3,^X34F12BB2
                 2C58DFF8 EBFB3FF6  03C0    235          .LONG   ^XEBFB3FF6,^X2C58DFF8   ; 9.38384792808987157552934621731767 5E-4
                 973C06EC F16BEA86  03C8    236          .LONG   ^XF16BEA86,^X973606EC
                 A39E2FEF 62E43FFC  03D0    237          .LONG   ^X62E43FFC,^XA39E2FEF   ; 4.33216987849965818385770075911360 3E-2
                 07E66730 93C7F357  03D8    238          .LONG   ^X93C7F357,^X07E66730
                 00000000 00000000  03E0    239          .LONG   ^X00000000,^X00000000   ; 0.0000000000000000000000000000000000E0
                 00000000 00000000  03E8    240          .LONG   ^X00000000,^X00000000
                          0000000F  03F0    241 HXPLN=<.-HXPTAB>/16                      ; No. of entries in table
                                    03F0    242
                                    03F0    243
                                    03F0    244 H_16LOG2_E:                              ; LOG2(E) * 16
                 B82F7652 71544005  03F0    245          .LONG   ^X71544005,^XB82F7652
                 D23AFDA0 7D0FE177  03F8    246          .LONG   ^X7D0FE177,^XD23AFDA0
                                    0400    247
                                    0400    248 H_LN2_OV_16_HI:
00006730 93C7F357 A39E2FEF 62E43FFC 0400    249          .OCTA   ^X0000673093C7F357A39E2FEF62E43FFC   ; Hi 95 bits of ln2/16
                                    0410    250
                                    0410    251 H_LN2_OV_16_LO:
069E16C5 4C5B9339 79A157A0 F97B3F96 0410    252          .OCTA   ^X069E16C54C5B933979A157A0F97B3F96   ; Low bits of ln2/16
                                    0420    253
```

```
                     0420   255          .SBTTL  MTH$HEXP - Standard H Floating EXP
                     0420   256
                     0420   257
                     0420   258 ;++
                     0420   259 ; FUNCTIONAL DESCRIPTION:
                     0420   260 ;
                     0420   261 ; EXP  - H floating point function
                     0420   262 ;
                     0420   263 ; Uses a Chebyshev approximation, with overhang on last step.
                     0420   264 ;
                     0420   265 ;
                     0420   266 ; CALLING SEQUENCE:
                     0420   267 ;
                     0420   268 ;       exponential.wh.v = MTH$HEXP(x.rh.r)
                     0420   269 ;
                     0420   270 ;               -or-
                     0420   271 ;
                     0420   272 ;       CALL MTH$HEXP(exponential.wh.r, x.rh.r)
                     0420   273 ;
                     0420   274 ;       Because an H-floating result cannot be expressed in 64 bits, it is
                     0420   275 ;       returned as the first argument, with the input parameter displaced
                     0420   276 ;       to the second argument, in accordance with the Procedure Calling
                     0420   277 ;       Standard.
                     0420   278 ;
                     0420   279 ; INPUT PARAMETERS:
                     0420   280
00000004             0420   281          LONG = 4                        ; Define longword multiplier
00000008             0420   282          x = 2 * LONG                    ; Contents of x is the argument
                     0420   283
                     0420   284 ; IMPLICIT INPUTS:      none
                     0420   285 ;
                     0420   286 ; OUTPUT PARAMETERS:
                     0420   287 ;
00000004             0420   288          exp = 1 * LONG                  ; Contents of exp is the result
                     0420   289
                     0420   290 ;       VALUE:  H floating exponential of the argument
                     0420   291 ;
                     0420   292 ; IMPLICIT OUTPUTS:     none
                     0420   293 ;
                     0420   294 ; SIDE EFFECTS:
                     0420   295 ;
                     0420   296 ; Signals: MTH$_FLOOVEMAT if X > 11355.83 with reserved operand in R0/R3
                     0420   297 ; (copied to the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                     0420   298 ; Associated message is: "FLOATING OVERFLOW IN MATH LIBRARY". Result is
                     0420   299 ; reserved operand -0.0 unless a user supplied (or any) error handler
                     0420   300 ; changes CHF$L_MCH_R0/R1.
                     0420   301 ;       MTH$_FLOUNDMAT if X =< -11356.52 and caller has hardware enable set.
                     0420   302 ; The result is set to +0.0. Associated message is: "FLOATING UNDERFLOW
                     0420   303 ; IN MATH LIBRARY"
                     0420   304 ;
                     0420   305 ; NOTE: This procedure disables floating point underflow, enable integer
                     0420   306 ; overflow, causes no floating overflow or other arithmetic traps, and
                     0420   307 ; preserves enables across the call.
                     0420   308 ;
                     0420   309 ;--
                     0420   310
                     0420   311
```

```
                         407C  0420   312          .ENTRY  MTH$HEXP, ACMASK        ; Standard call-by-reference entry
                               0422   313                                          ; Disable DV (and FU), enable IV
                               0422   314          MTH$FLAG_JACKET                 ; Flag that this is a jacket procedure
                               0422
    6D  00000000'GF    9E      0422                MOVAB   G^MTH$$JACKET_HND, (FP)
                               0429                                                ; set handler address to jacket
                               0429                                                ; handler
                               0429
                               0429   315                                          ; in case of an error in special JSB
                               0429   316                                          ; routine
          50    08 BC  70FD    0429   317          MOVH    @x(AP), R0              ; R0/R3 = user's arg
                06     10     042E   318          BSBB    MTH$HEXP_R6             ; R0/R3 = special HEXP(R0/R3)
          04 BC 50     7DFD    0430   319          MOVQ    R0, @exp?AP)            ; Store result in first argumen t
                         04    0435   320          RET                             ; Return to caller
```

```
                                0436   322          .SBTTL  MTH$HEXP_R6 - Special HEXP routine
                                0436   323
                                0436   324 ; Special HEXP - used by the standard, and direct interfaces.
                                0436   325 ;
                                0436   326 ; CALLING SEQUENCE:
                                0436   327 ;       save anything needed in R0:R6
                                0436   328 ;       MOVH    ...,R0                  ; Input in R0/R3
                                0436   329 ;       JSB     MTH$HEXP_R6
                                0436   330 ;       return with result in R0/R3
                                0436   331
                                0436   332 ; Note: This routine is written to avoid causing any integer overflows,
                                0436   333 ; floating overflows, or floating underflows or divide by 0 conditions,
                                0436   334 ; whether enabled or not.
                                0436   335 ;
                                0436   336 ; REGISTERS USED:
                                0436   337 ;       R0/R3 - H-floating argument, then H-floating result
                                0436   338
                                0436   339
                                0436   340 MTH$HEXP_R6::                           ; Special HEXP routine
                                0436   341 ;+
                                0436   342 ;   The  preliminary  test  for overflow works as follows:  First, the sign is
                                0436   343 ;   taken away, leaving just the biased exponent.  Then,  16384-4  (bias-4) is
                                0436   344 ;   subtracted, leaving  an  exponent  biased by 4.  This rebiased exponent is
                                0436   345 ;   compared against 18.  The comparison can have 3 outcomes.  If the rebiased
                                0436   346 ;   exponent is now negative, this means that the true expcnent is < -4 - this
                                0436   347 ;   is a BLSSU test.  If the rebiased exponent is positive,  but  greater than
                                0436   348 ;   18, then  the  actual  value  is  greater  than 16384, which is guaranteed
                                0436   349 ;   overflow or underflow, depending on the sign of X -  this  is a BLSS test.
                                0436   350 ;   Otherwise,  X  is  somewhere in the range for the standard evaluation, and
                                0436   351 ;   flow continues.
                                0436   352 ;
                                0436   353 ;-
      54    50    8000 8F   AB  0436   354          BICW3   #^X8000, R0, R4         ; Preliminary test for over/underflow
                                043C   355                                          ; R4 = exponent bits only
      55    54    3FFC 8F   A3  043C   356          SUBW3   #^X3FFC, R4, R5         ; R5 = 4 + unbiased exponent
                    55   12   B1  0442   357          CMPW    #18, R5                 ; Unsigned compare with 18
                         71   1F  0445   358          BLSSU   SMTST                   ; To more tests if LSSU
                                0447   359                                          ;  else, -4 < unbiased exp < 15
                                0447   360                                          ;  no exceptions in EMODH or APPROX
                                0447   361
            400C 8F     54   B1  0447   362          CMPW    R4, #^X400C             ; Check for loss of significance in
                                044C   363                                          ; EMOD ( !X! >= 2**11)
                         29   19  044C   364          BLSS    EVAL                    ; No loss of significance
                                044E   365
                                044E   366 ;
                                044E   367 ; !X! >= 2**4.  EMOD will lose significance so the interger and fractional
                                044E   368 ; parts of X*16/ln2 must be obtained in seperate steps.
                                044E   369 ;
    7E    50     9E AF 65FD  044E   370          MULH3   H_16LOG2_E, R0, -(SP)   ; Get integer part of X*16/ln2 in
                56   6E 6AFD  0454   371          CVTHL   (SP), R6                ;    R6 (=I+J) as a longword and in
                6E   56 6EFD  0458   372          CVTLH   R6, (SP)                ;    (SP) in H format
                                045C   373                                          ; Get fraction part of X*16/ln2 =
    7E    6E     A0 AF 65FD  045C   374          MULH3   H_LN2_OV_16_HI, (SP), -(SP)
                50   8E 62FD  0462   375          SUBH2   (SP)+, R0               ;    16/ln2*[ X - (I+J)*ln2/16 ]
          6E     A6 AF 64FD  0466   376          MULH2   H_LN2_OV_16_LO, (SP)    ;    in R0/R1.
                50   8E 62FD  046B   377          SUBH2   (SP)+, R0               ;
          50    FF7C CF 64FD  046F   378          MULH2   H_16LOG2_E, R0          ;
```

```
                                0B   11  0475   379          BRB     APPROX                          ;
                                                0477   380
   56   50   7D12 8F   FF74 CF 74FD  0477   381 EVAL:    EMODH   H_16LOG2_E, #X_16LOG2E, R0, R6, R0
                                50      0481
                                                0482   382                                          ; Get X*16*LOG2(E) with
                                                0482   383                                          ;  integer part in R6 (=16I+J)
                                                0482   384                                          ;  fraction in R0/R3
                                                0482   385
          FE77 CF    0E   50 75FD  0482   386 APPROX: POLYH   R0,#HXPLN-1,HXPTAB                    ; Use Chebyshev series
                                                0489   387                                          ;  with last coefficient 0
                                                0489   388                                          ;  so that last ADDH has overhang
                                                0489   389
        54   56   FFFFFFF0 8F   CB  0489   390          BICL3   #-16, R6, R4                        ; R4 = J
             50   FB69 CF44 64FD  0491   391          MULH2   TABHI[R4], R0                       ; Else MUL by 2**(J/16)
             50   FC62 CF44 60FD  0498   392          ADDH2   TABLO[R4], R0                       ; Add in LO of 2**(J/16)
             50   FB5B CF44 60FD  049F   393          ADDH2   TABHI[R4], R0                       ; And then HI of 2**(J/16)
                                                04A6   394
                     56   0F   CA  04A6   395          BICL    #15, R6                             ; R6 = I
                          0C   13  04A9   396          BEQL    20$                                 ; If I=0, then done
                56   56   FC 8F   78  04AB   397          ASHL    #-4, R6, R6                         ; Position I for addition
                     50   56   C0  04B0   398          ADDL2   R6, R0                              ; Add I to exponent.
                                                04B3   399                                          ; MUL by 2**I by exponent addition
                          50   B5  04B3   400          TSTW    R0                                  ; Test for over/underflow
                          42   15  04B5   401          BLEQ    EXCEPT                              ; See what exception is if neg or = 0
                               05  04B7   402 20$:     RSB                                         ; Otnerwise return result in R0
                                                04B8   403
                                                04B8   404 SMTST:
                          14   19  04B8   405          BLSS    20$                                 ; Exception if exp+4 > 18
                3F8F 8F   54   B1  04BA   406          CMPW    R4, #^X3F8F                         ; Eliminate underflow from APPROX1
                          08   19  04BF   407          BLSS    10$                                 ; Bypass if E**ARG = 1
                                                04C1   408
                                                04C1   409 ;+
                                                04C1   410 ; Use Chebyshev series for small arg
                                                04C1   411 ;-
                                                04C1   412
   FD38 CF    0F   50 75FD  04C1   413          POLYH   R0,#HXPLN1-1,HXPTB1                 ;Use Chebyshev series
                                                04C8   414                                          ;  last term is 1; this will
                                                04C8   415                                          ;  give desired overhang.
                     05  04C8   416          RSB                                                 ; Answer is OK, return
                                                04C9   417
             50   08 70FD  04C9   418 10$:     MOVH    S^#1.0, R0                                  ; E**X is 1.0, store it
                     05  04CD   419          RSB                                                 ; And return
                                                04CE   420
                                                04CE   421
                                                04CE   422 ;
                                                04CE   423 ; Handlers for software detected over/underflow conditions follow
                                                04CE   424 ;
                     50 73FD  04CE   425 20$:     TSTH    R0                                  ; If big ARG > 0 goto OVERFLOW
                     2A   18  04D1   426          BGEQ    OVER
                                                04D3   427 ;
                                                04D3   428 ; Underflow; if user has FU set, signal error.  Always return 0.0
                                                04D3   429 ;
                                                04D3   430 UNDER:
                     54   DC  04D3   431          MOVPSL  R4                                  ; R4 = user's or jacket routine's PSL
 00000000'GF         00   FB  04D5   432          CALLS   #0, G^MTH$$JACKET_TST               ; R0 = TRUE if JSB from jacket routine
               04    50   E9  04DC   433          BLBC    R0, 10$                             ; Branch if user did JSB
          54   04    AD   3C  04DF   434          MOVZWL  SF$W_SAVE_PSW(FP), R4               ; Get user PSL saved by CALL
```

```
                    50   7C  04E3    435 10$:    CLRQ    R0                          ; R0 = result, LIB$SIGNAL will save in
                         04E5    436                                                 ; CHF$L_MCH_R0/R1 so any handler can fixup
                    52   7C  04E5    437         CLRQ    R2                          ; ...
                0D 54    06  E1  04E7    438         BBC     #6, R4, 20$             ; Has user enabled floating underflow?
                         6E  DD  04EB    439         PUSHL   (SP)                    ; Yes, return PC from special routine
                7E   00'8F   9A  04ED    440         MOVZBL  #MTH$K_FLOUNDMAT, -(SP) ; Trap code for hardware floating underflow
                         04F1    441                                                 ; Convert to MTH$_FLOUNDMAT (32-bit VAX-11
                         04F1    442                                                 ; Exception code)
        00000000'GF      02  FB  04F1    443         CALLS   #2, G^MTH$$SIGNAL       ; Signal (condition, PC)
                         05  04F8    444 20$:    RSB                                 ; Return
                         04F9    445
                         04F9    446 EXCEPT:
                    56   D5  04F9    447         TSTL    R6                          ; Test sign of I; if I < 0
                    D6   19  04FB    448         BLSS    UNDER                       ; Go to underflow handler
                         04FD    449 ;
                         04FD    450 ; Signal floating overflow, return reserved operand, -0.0
                         04FD    451 ;
                         04FD    452 OVER:                                           ; Else process for overflow
                         6E  DD  04FD    453         PUSHL   (SP)                    ; Return PC from special routine
                7E   00'8F   9A  04FF    454         MOVZBL  #MTH$K_FLOOVEMAT, -(SP) ; Hardware floating overflow
            50   01   0F  79  0503    455         ASHQ    #15, #1, R0                 ; R0/R3 = result = reserved operand
                         0507    456                                                 ; -0.0. R0/0 will be copied to
                         0507    457                                                 ; Signal mechanism vector (CHF$L_MCH_R0/R1)
                         0507    458                                                 ; So can be fixed up by any error
                         0507    459                                                 ; Handler
                    52   7C  0507    460         CLRQ    R2                          ; ...
        00000000'GF      02  FB  0509    461         CALLS   #2, G^MTH$$SIGNAL       ; Signal (condition, PC)
                         05  0510    462         RSB                                 ; Return - R0 restored from CHF$L_MCH_R0/R1
                         0511    463
                         0511    464
                         0511    465         .END                                    ; End of module MTH$HEXP
```

```
ACMASK          =  0000407C
APPROX             00000482 R      02
EVAL               00000477 R      02
EXCEPT             000004F9 R      02
EXP             =  00000004
HXPLN           =  0000000F
HXPLN1          =  00000010
HXPTAB             00000300 R      02
HXPTB1             00000200 R      02
H_16LOG2_E         000003F0 R      02
H_LN2_OV_16_HI     00000400 R      02
H_LN2_OV_16_LO     00000410 R      02
LONG            =  00000004
MTH$$JACKET_HND    ********   X    02
MTH$$JACKET_TST    ********   X    00
MTH$$SIGNAL        ********   X    00
MTH$HEXP           00000420 RG     02
MTH$HEXP_R6        00000436 RG     02
MTH$K_FLOOVEMAT    ********   X    00
MTH$K_FLOUNDMAT    ********   X    00
OVER               000004FD R      02
SF$W_SAVE_PSW   =  00000004
SMTST              000004B8 R      02
TABHI              00000000 R      02
TABLO              00000100 R      02
UNDER              000004D3 R      02
X               =  00000008
X_16LOG2E       =  00007D12
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_MTH$CODE	00000511 (1297.)	02 (2.)	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	00:00:00.44
Command processing	116	00:00:00.72	00:00:06.89
Pass 1	133	00:00:02.23	00:00:07.48
Symbol table sort	0	00:00:00.03	00:00:00.03
Pass 2	100	00:00:01.15	00:00:04.85
Symbol table output	4	00:00:00.03	00:00:00.18
Psect synopsis output	2	00:00:00.03	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	386	00:00:04.30	00:00:19.94

```
The working set limit was 1050 pages.
9258 bytes (19 pages) of virtual memory were used to buffer the intermediate code.
```

There were 10 pages of symbol table space allocated to hold 56 non-local and 5 local symbols.
525 source lines were read in Pass 1, producing 15 object records in Pass 2.
9 pages of virtual memory were used to define 8 macros.

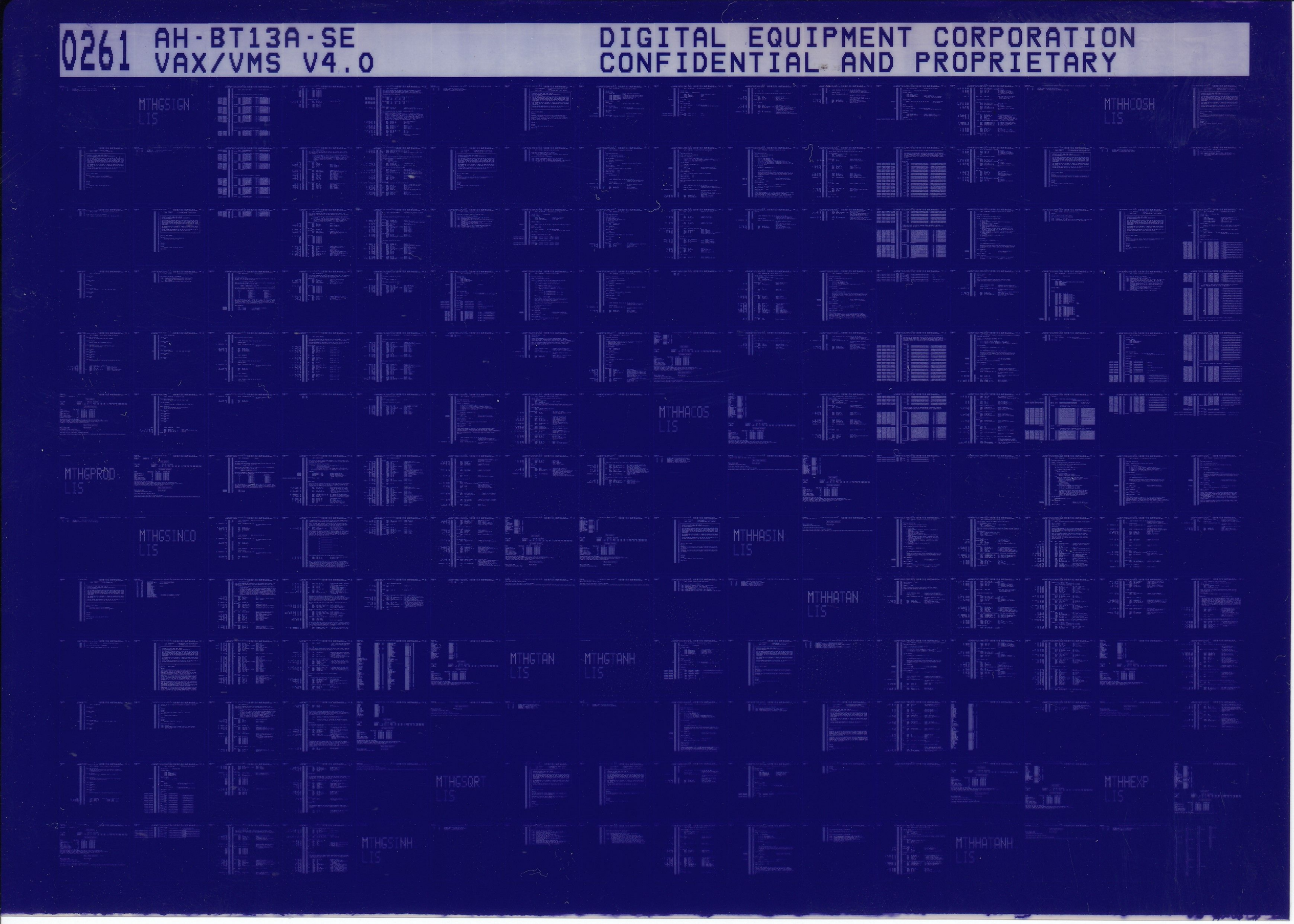
+----------------------------+
! Macro library statistics !
+----------------------------+

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	4

88 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

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



MTHIISIGN
LIS

MTHHFLOOR
LIS

MTHHSIGN
LIS

MTHHMIN
LIS

MTHHLOG
LIS

MTHHTAN
LIS

MTHIIONNT
LIS

MTHIIHNNT
LIS

MTHHSQRT
LIS

MTHIMAX0
LIS

MTHHNINT
LIS

MTHHSINH
LIS

MTHHTANH
LIS

MTHHINT
LIS

MTHHMAX1
LIS

MTHHSINCO
LIS

MTHHMOD
LIS

MTHIIGNNT
LIS