```
MMM         MMM  TTTTTTTTTTTTTTT  HHH         HHH  RRRRRRRRRRRR     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        HHHHHHHHHHHHHHH  RRRRRRRRRRRR           TTT        LLL
MMM         MMM        TTT        HHHHHHHHHHHHHHH  RRRRRRRRRRRR           TTT        LLL
MMM         MMM        TTT        HHHHHHHHHHHHHHH  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        LLLLLLLLLLLLLLL
MMM         MMM        TTT        HHH         HHH  RRR         RRR        TTT        LLLLLLLLLLLLLLL
MMM         MMM        TTT        HHH         HHH  RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**MTHGATANH


```
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG    AAAAAA    TTTTTTTTTT    AAAAAA    NN      NN  HH      HH
MM      MM  TTTTTTTTTT  HH      HH    GGGGGGGG    AAAAAA    TTTTTTTTTT    AAAAAA    NN      NN  HH      HH
MMMM  MMMM      TT      HH      HH  GG          AA      AA      TT      AA      AA  NN      NN  HH      HH
MMMM  MMMM      TT      HH      HH  GG          AA      AA      TT      AA      AA  NN      NN  HH      HH
MM  MM  MM      TT      HH      HH  GG          AA      AA      TT      AA      AA  NNNN    NN  HH      HH
MM  MM  MM      TT      HH      HH  GG          AA      AA      TT      AA      AA  NNNN    NN  HH      HH
MM      MM      TT      HHHHHHHHHH  GG          AA      AA      TT      AA      AA  NN  NN  NN  HHHHHHHHHH
MM      MM      TT      HHHHHHHHHH  GG          AA      AA      TT      AA      AA  NN  NN  NN  HHHHHHHHHH
MM      MM      TT      HH      HH  GG  GGGGGG  AAAAAAAAAA      TT      AAAAAAAAAA  NN    NNNN  HH      HH
MM      MM      TT      HH      HH  GG  GGGGGG  AAAAAAAAAA      TT      AAAAAAAAAA  NN    NNNN  HH      HH
MM      MM      TT      HH      HH  GG      GG  AA      AA      TT      AA      AA  NN      NN  HH      HH      ....
MM      MM      TT      HH      HH  GG      GG  AA      AA      TT      AA      AA  NN      NN  HH      HH      ....
MM      MM      TT      HH      HH    GGGGGG    AA      AA      TT      AA      AA  NN      NN  HH      HH      ....
MM      MM      TT      HH      HH    GGGGGG    AA      AA      TT      AA      AA  NN      NN  HH      HH      ....

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
VAX

Mac

_$2

0 (

The

MA(

Table of contents

```
0000          .TITLE  MTH$GATANH      ; G-floating Precision Hyperbolic Arctangent
0000          .IDENT /2-002/          ; File: MTHGATANH.MAR  Edit: PDG2002
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000    ;
0000    ; FACILITY: MATH LIBRARY
0000    ;++
0000    ; ABSTRACT:
0000    ;
0000    ; MTH$GATANH returns the G-floating precision hyperbolic arctangent of the
0000    ; G-floating precision argument.  The call is standard call-by-reference.
0000    ;
0000    ;--
0000    ;
0000    ; VERSION: 2
0000    ;
0000    ; HISTORY:
0000    ; AUTHOR:
0000    ;       Peter D Gilbert, 23-Jul-81: Version 2
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;
```

```
0000          .SBTTL  HISTORY ; Detailed Current Edit History
0000
0000   ; VERSION 1
0000   ;
0000   ; Edit History for Version 01 of MTH$GATANH
0000   ;
0000   ; 01-0  Original        July 1981
0000   ;
0000   ; Edit History for Version 02 of MTH$HATANH
0000   ;
0000   ; 2-000   Original      July 1981
0000   ; 2-001 - Change MOVZBL to CVTBL when accessing MTH$$AB_ALOG_V.  PDG 2-Dec-1981
0000   ; 2-002 - Change RSB to RET after error exit.  PDG 6-Jan-1981
```

```
          0000          .SBTTL  DECLARATIONS    ; Declarative Part of Module
          0000
          0000  ;
          0000  ; INCLUDE FILES:        MTHJACKET.MAR
          0000  ;
          0000
          0000  ;
          0000  ; EXTERNAL SYMBOLS:
          0000  ;
          0000          .DSABL  GLOBAL
          0000          .SHOW   BINARY,CALLS,CONDITIONALS,DEFINITIONS,EXPANSIONS
          0000          .EXTRN  MTH$K_INVARGMAT
          0000          .EXTRN  MTH$$SIGNAL
          0000          .EXTRN  MTH$$AB_ALOG_V
          0000          .EXTRN  MTH$$AB_G_FHI
          0000
          0000  ;
          0000  ; EQUATED SYMBOLS:
          0000  ;
          0000
          0000  ;
          0000  ; MACROS:
          0000  ;
          0000
          0000          .MACRO  OPDEF   X, OP, SH
          0000          .OPDEF  ADDX    ^X00@SH+OP,R'X,M'X
          0000          .OPDEF  ADDX3   ^X01@SH+OP,R'X,R'X,W'X
          0000          .OPDEF  SUBX    ^X02@SH+OP,R'X,M'X
          0000          .OPDEF  SUBX3   ^X03@SH+OP,R'X,R'X,W'X
          0000          .OPDEF  MULX    ^X04@SH+OP,R'X,M'X
          0000          .OPDEF  MULX3   ^X05@SH+OP,R'X,R'X,W'X
          0000          .OPDEF  DIVX    ^X06@SH+OP,R'X,M'X
          0000          .OPDEF  DIVX3   ^X07@SH+OP,R'X,R'X,W'X
          0000          .OPDEF  CVTWX   ^X0D@SH+OP,RW,W'X
          0000          .OPDEF  POLYX   ^X15@SH+OP,R'X,RW,AB
          0000          .OPDEF  MOVX    ^X007D,RQ,WQ            ; MOVQ
          0000          .OPDEF  MOVAX   ^X007E,AQ,WL            ; MOVAQ
          0000          .ENDM
          0000
          0000          OPDEF   G, <^X40FD>, 8
          0000          .OPDEF  ADDX    ^X00@8+^X40FD,RG,MG
          0000          .OPDEF  ADDX3   ^X01@8+^X40FD,RG,RG,WG
          0000          .OPDEF  SUBX    ^X02@8+^X40FD,RG,MG
          0000          .OPDEF  SUBX3   ^X03@8+^X40FD,RG,RG,WG
          0000          .OPDEF  MULX    ^X04@8+^X40FD,RG,MG
          0000          .OPDEF  MULX3   ^X05@8+^X40FD,RG,RG,WG
          0000          .OPDEF  DIVX    ^X06@8+^X40FD,RG,MG
          0000          .OPDEF  DIVX3   ^X07@8+^X40FD,RG,RG,WG
          0000          .OPDEF  CVTWX   ^X0D@8+^X40FD,RW,WG
          0000          .OPDEF  POLYX   ^X15@8+^X40FD,RG,RW,AB
          0000          .OPDEF  MOVX    ^X007D,RQ,WQ            ; MOVQ
          0000          .OPDEF  MOVAX   ^X007E,AQ,WL            ; MOVAQ
          0000
          0000
00000007  0000          F_EXP = 7       ; Bit offset to exponent
00000004  0000          X_EXP = 4       ; Bit offset to exponent
          0000
```

```
    0000   ;
    0000   ; PSECT DECLARATIONS:
    0000   ;
00000000         .PSECT  _MTH$CODE       PIC,SHR,LONG,EXE,NOWRT
    0000                                         ; program section for math routines
    0000   ;
    0000   ; OWN STORAGE:  none
    0000   ;
```

```
                            0000  ; CONSTANTS:
                            0000  ;
                            0000
                  000001FC  0000          ACMASK = ^M<R2,R3,R4,R5,R6,R7,R8>
                            0000                                          ; register entry mask and integer
                            0000                                          ; overflow enable
                            0000
                            0000  ;+  The "16" in the constants is used to shift the unbiased exponent
                            0000  ;   right 4 places so that the rightmost bit is at bit 0.
                            0000  ;-
                            0000
         2800FEF6 2E423FC6  0000  LN2_HI: .QUAD   ^X2800FEF62E423FC6      ; (Hi 42 bits of ln2)/16
         F1DAD5E4 47BC3DA0  0008  LN2_LO: .QUAD   ^XF1DAD5E447BC3DA0      ; (Low bits of ln2)/16
                            0010
                            0010  LOGTAB1:                                ; Constants for q(z).  Generated using
                            0010                                          ; eq. 6.3.10 of Hart et. al. (sin(2a)
                            0010                                          ; = 1/32)
         A8981E57 81CD3FDC  0010          .QUAD   ^XA8981E5781CD3FDC      ; C8 =   0.11135560980588577
         38EFC0D0 0802BFE0  0018          .QUAD   ^X38EFC0D00802BFE0      ; C7 =  -0.12524446882930060
         C9769148 49223FE2  0020          .QUAD   ^XC976914849223FE2      ; C6 =   0.14285690397225509
         BBAC9487 5553BFE5  0028          .QUAD   ^XBBAC94875553BFE5      ; C5 =  -0.16666645767642529
         B92699D1 99993FE9  0030          .QUAD   ^XB92699D199993FE9      ; C4 =   0.20000000010208757
         0A540014 0000BFF0  0038          .QUAD   ^X0A5400140000BFF0      ; C3 =  -0.25000000007290635
         54155555 55553FF5  0040          .QUAD   ^X5415555555553FF5      ; C2 =   0.33333333333331555
         FF60FFFF FFFFBFFF  0048          .QUAD   ^XFF60FFFFFFFFBFFF      ; C1 =  -0.49999999999999112
         00000000 00000000  0050          .QUAD   ^X0000000000000000      ; C0 =   0.00000000000000000
                  00000008  0058  LOGLEN1 = .-LOGTAB1/8 - 1               ; no. of floating point entries
                            0058
                            0058
                            0058  LOGTAB2:                                ; Constants for p(z*z). Generated using
                            0058                                          ; eq. 6.3.11 of Hart et. al. (sin(2a) =
                            0058                                          ; (b - 1)/(b + 1) where b = 2**(1/7))
         B117401D 6E163FE7  0058          .QUAD   ^XB117401D6E163FE7      ; C5 =  0.18304708605445149 7
         0BA587C0 71A73FEC  0060          .QUAD   ^X0BA587C071A73FEC      ; C4 =  0.22221845749308247 2
         C30B9839 49243FF2  0068          .QUAD   ^XC30B983949243FF2      ; C3 =  0.28571429124626551 7
         839E9998 99993FF9  0070          .QUAD   ^X839E999899993FF9      ; C2 =  0.39999999999604962 7
         55605555 55554005  0078          .QUAD   ^X5560555555554005      ; C1 =  0.66666666666666785 1
         00000000 00004020  0080          .QUAD   ^X0000000000004020      ; C0 =  2.00000000000000000 0
                  00000005  0088  LOGLEN2 = .-LOGTAB2/8 - 1
```

```
          0088        .SBTTL  MTH$GATANH      - G-floating Precision Hyperbolic Arctangent
          0088
          0088  ;++
          0088  ; FUNCTIONAL DESCRIPTION:
          0088  ;
          0088  ; GATANH  - G-floating precision floating point function
          0088  ;
          0088  ; GATANH(X) is computed using the following approximation technique:
          0088  ;
          0088  ;       If |X| >= 1.0, error.  Otherwise
          0088  ;
          0088  ;       Let (1+X)/(1-X) = f * (2**n), where 1/2 <= f < 1
          0088  ;
          0088  ;       If n is greater than or equal to 1 then
          0088  ;           set N = n - 1 and F1 = 2*f.
          0088  ;       Else
          0088  ;           set N = n and F = f.
          0088  ;
          0088  ;       If |F - 1| < 2**-5 then
          0088  ;           2*atanh(X) =  N*ln(2) + W + W*P(W),
          0088  ;               where W = ((1+F)/(1-F))*2**N - 1,
          0088  ;               and P is a polynomial of degree F=5,D=9.
          0088  ;       Else
          0088  ;           Obtain FHI (roughly equal to F) from table lookup.
          0088  ;           2*atanh(X) = ln((1+X)/(1-X)) = N*ln(2) + ln(FHI) + Z*Q(Z*Z),
          0088  ;               where Q is a polynomial of degree F=2,D=5,
          0088  ;               where Z = (F - FHI)/(F + FHI)
          0088  ;               where F = (2**-N)*(1+X)/(1-X)
          0088  ;           Z is computed by:
          0088  ;               Z = (X-D)/(1-X*D)
          0088  ;               where Y = FHI*2**N
          0088  ;               where D = (Y-1)/(Y+1)
          0088  ;           Note that Z may be computed in a variety of ways:
          0088  ;               Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
          0088  ;               Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
          0088  ;               Z = [1 - Y + X + X*Y]/[1 + Y + X - X*Y]
          0088  ;               Z = [(1-Y) + X*(1+Y)]/[(1+Y) + X*(1-Y)]
          0088  ;
          0088  ;       NOTE:   The quantities ln(A=FHI) and ln2 are used in the above
          0088  ;               equations in two parts - a high part (containing the
          0088  ;               high order bits) and a low part (containing the low
          0088  ;               order bits.  In the code the high and low parts of the
          0088  ;               constants are indicated by a _HI and _LO suffix respec-
          0088  ;               tively.  The values were chosen such that N*LN2_HI +
          0088  ;               LN_FHI_HI is exactly representable.
          0088  ;
          0088  ; CALLING SEQUENCE:
          0088  ;
          0088  ;       gatanh.wg.v = MTH$GATANH(x.rg.r)
          0088  ;
          0088  ; INPUT PARAMETERS:
          0088
00000004  0088          X = 4                           ; Contents of x is the argument
          0088
          0088  ; IMPLICIT INPUTS:      none
          0088  ;
          0088  ; OUTPUT PARAMETERS:
```

```
                              0088  ;
                              0088  ;       VALUE:  G-floating precision hyperbolic arctangent of the argument
                              0088  ;
                              0088  ; IMPLICIT OUTPUTS:     none
                              0088  ;
                              0088  ; COMPLETION CODES:     none
                              0088  ;
                              0088  ; SIDE EFFECTS:
                              0088  ;
                              0088  ; Signals: MTH$K_INVARGMAT if !X! >= 1.0 with reserved operand in R0 (copied to
                              0088  ; the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                              0088  ; Associated message is: "Floating overflow in math library". Result is
                              0088  ; reserved operand -0.0 unless a user supplied (or any) error handler changes
                              0088  ; CHF$L_MCH_R0/R1.
                              0088  ;
                              0088  ; NOTE: This procedure disables floating point underflow and integer
                              0088  ; overflow, causes no floating overflow or other arithmetic traps, and
                              0088  ; preserves enables across the call.
                              0088  ;
                              0088  ; Note: This routine is written to avoid causing any integer overflows,
                              0088  ; floating overflows, or floating underflows or divide by 0 conditions,
                              0088  ; whether enabled or not.
                              0088  ;
                              0088  ;---
                              0088
              016A    31      0088  ERR:    BRW     ERROR
                              008B
                      01FC    008B          .ENTRY  MTH$GATANH, ACMASK      ; standard call-by-reference entry
                              008D                                          ; disable DV (and FU), enable IV
      50    04 BC     7D      008D          MOVX    @X(AP), R0              ; R0 = arg
                              0091
         56    50     7D      0091          MOVX    R0, R6
     52    08    56 43FD      0094          SUBX3   R6, S^#1.0, R2          ; R2 = 1-X
               ED     15      0099          BLEQ    ERR                     ; ATANH(X) is not defined for X>=1
         56    08   40FD      009B          ADDX        S^#1.0, R6
               E7     15      009F          BLEQ    ERR                     ; ATANH(X) is not defined for X<=-1
         52    52   33FD      00A1          CVTGF   R2, R2
         56    56   33FD      00A5          CVTGF   R6, R6
         56    52     46      00A9          DIVF2   R2, R6                  ; R6 = approximation to (1+X)/(1-X)
 58    00000000'GF    9E      00AC          MOVAB   G^MTH$$AB_ALOG_V, R8
         58    68     C0      00B3          ADDL2   (R8), R8                ; R8 = address of ALOG table
  55   56  007F 8F    AB      00B6          BICW3   #1@F_EXP-1, R6, R5      ; R5 = Biased exponent
     55    4000 8F    A2      00BC          SUBW    #^X4000, R5             ; R5 = Unbiased exponent
               76     15      00C1          BLEQ    NEG_EXP                 ; Branch to processing for n=<0
                              00C3
     55    0080 8F    A2      00C3          SUBW    #1@F_EXP, R5            ; Exponent is positive, R5 = N = n - 1
         56    55     A2      00C8          SUBW    R5, R6                  ; R6 = F = 2f
         56    56     9A      00CB          MOVZBL  R6, R6                  ; R6 = index into ALOG table
           00000003           00CE          .IF     NE, F_EXP-X_EXP
         55    08     A6      00CE          DIVW2   #1@<F_EXP-X_EXP>, R5    ; Shift R5 to scale X-floating
                              00D1          .ENDC
         7E    55   4DFD      00D1          CVTWX   R5, -(SP)               ; Push N onto the stack
       58    6846     98      00D5          CVTBL   (R8)[R6], R8            ; R8 = offset into FHI tables
               6F     19      00D9          BLSS    LN_1_PLUS_W             ; Branch to handle F close to 1
58   00000000'GF48    7E      00DB          MOVAX   G^MTH$$AB_G_FHI[R8], R8 ; R8 = Address of FHI
         56    88     7D      00E3          MOVX    (R8)+, R6               ; R6 = FHI
                              00E6  ;
```

```
                           00E6  ; Compute Z = (F - FHI)/(F + FHI)
                           00E6  ;                   Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
                           00E6  ;                   Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
                           00E6  ;         where Y = FHI*2**N, roughly equal to (1+X)/(1-X)
                           00E6  ;
            56    55   A0  00E6            ADDW     R5, R6                   ; R6 = FHI * 2**N = SFHI
      52    08    56 43FD  00E9            SUBX3    R6, S^#1.0, R2           ; R2 = 1 - SFHI
            56    08 40FD  00EE            ADDX     S^#1.0, R6               ; R6 = 1 + SFHI
            52    56 46FD  00F2            DIVX     R6, R2                   ; R2 = (1-SFHI)/(1+SFHI) = D
      56    52    50 41FD  00F6            ADDX3    R0, R2, R6               ; R6 = D + X
            52    50 44FD  00FB            MULX     R0, R2                   ; R2 = D * X
            52    08 40FD  00FF            ADDX     S^#1.0, R2               ; R2 = 1 + D*X
            56    52 46FD  0103            DIVX     R2, R6                   ; R6 = (D+X)/(1+D*X) = Z
                           0107  ;
                           0107  ; Compute Z**2, P(Z**2) and Z*P(Z**2)
                           0107  ;
      50    56    56 45FD  0107            MULX3    R6, R6, R0               ; R0 = Z**2
 FF45 CF    05    50 55FD  010C            POLYX    R0, #LOGLEN2, LOGTAB2    ; R0 = P(Z**2)
            50    56 44FD  0113            MULX     R6, R0                   ; R0 = Z*P(Z**2)
                           0117  ;
                           0117  ; Compute B = N*LN2_LO + LN_FHI_LO + Z*P(Z*Z)
                           0117  ;
 52    FEEB CF    6E 45FD  0117            MULX3    (SP), LN2_LO, R2         ; R2 = N*LN2_LO
            52    88 40FD  011E            ADDX     (R8)+, R2                ; R2 = N*LN2_LO + LN_FHI_LO
            50    52 40FD  0122            ADDX     R2, R0                   ; R0 = B
                           0126  ;
                           0126  ; Compute A = N*LN2_HI + LN_FHI_HI and ALOG(X)
                           0126  ;
 52    FED4 CF    8E 45FD  0126            MULX3    (SP)+, LN2_HI, R2        ; R2 = N*LN2_HI
            52    68 40FD  012D            ADDX     (R8), R2                 ; R2 = A = N*LN2_HI + LN_FHI_HI
            50    52 40FD  0131            ADDX     R2, R0                   ; R0 = A + B = ALOG(X)
            50    10   A2  0135            SUBW2    #1@X_EXP, R0             ; Divide by 2
                       04  0138            RET
                           0139
                           0139  NEG_EXP:
            56    55   A2  0139            SUBW     R5, R6                   ; R6 = F = 2f
            56    56   9A  013C            MOVZBL   R6, R6                   ; R6 = index into ALOG table
              00000003     013F            .IF      NE, F_EXP-X_EXP
            55    08   A6  013F            DIVW2    #1@<F_EXP-X_EXP>, R5     ; Shift R5 to scale X-floating
                           0142            .ENDC
            7E    55 4DFD  0142            CVTWX    R5, -(SP)                ; Push N onto the stack
         58     6846   98  0146            CVTBL    (R8)[R6], R8             ; R8 = offset into FHI tables
                           014A  LN_1_PLUS_W:
                  5E   19  014A            BLSS     LN_1_PLUS                ; Branch to handle F close to 1
 58      00000000'GF48 7E  014C            MOVAX    G^MTH$$AB_G_FHI[R8], R8  ; R8 = Address of FHI
            56    68   7D  0154            MOVX     (R8), R6                 ; R6 = FHI
                           0157  ;
                           0157  ; Compute Z = (F - FHI)/(F + FHI)
                           0157  ;                   Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
                           0157  ;                   Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
                           0157  ;         where Y = FHI*2**N, roughly equal to (1+X)/(1-X)
                           0157  ;
            56    55   A0  0157            ADDW     R5, R6                   ; R6 = FHI * 2**N = SFHI
      52    08    56 43FD  015A            SUBX3    R6, S^#1.0, R2           ; R2 = 1 - SFHI
            56    08 40FD  015F            ADDX     S^#1.0, R6               ; R6 = 1 + SFHI
            52    56 46FD  0163            DIVX     R6, R2                   ; R2 = (1-SFHI)/(1+SFHI) = D
      56    52    50 41FD  0167            ADDX3    R0, R2, R6               ; R6 = D + X
```

```
              52     50 44FD  016C           MULX    R0, R2                  ; R2 = D * X
              52     08 40FD  0170           ADDX    S^#1.0, R2              ; R2 = 1 + D*X
              56     52 46FD  0174           DIVX    R2, R6                  ; R6 = (D+X)/(1+D*X) = Z
                              0178      ;
                              0178      ; Compute Z**2, P(Z**2) and Z*P(Z**2)
                              0178      ;
          50  56     56 45FD  0178           MULX3   R6, R6, R0              ; R0 = Z**2
     FED4 CF  05     50 55FD  017D           POLYX   R0, #LOGLEN2, LOGTAB2   ; R0 = P(Z**2)
              50     56 44FD  0184           MULX    R6, R0                  ; R0 = Z*P(Z**2)
                              0188      ;
                              0188      ; Compute B = N*LN2_LO + LN_FHI_LO + Z*P(Z*Z)
                              0188      ;
     52   FE7A CF    6E 45FD  0188           MULX3   (SP), LN2_LO, R2        ; R2 = N*LN2_LO
              52     78 40FD  018F           ADDX    -(R8), R2               ; R2 = N*LN2_LO + LN_FHI_LO
              50     52 40FD  0193           ADDX    R2, R0                  ; R0 = B
                              0197      ;
                              0197      ; Compute A = N*LN2_HI + LN_FHI_HI and ALOG(X)
                              0197      ;
     52   FE63 CF    8E 45FD  0197           MULX3   (SP)+, LN2_HI, R2       ; R2 = N*LN2_HI
              52     78 42FD  019E           SUBX    -(R8), R2               ; R2 = A = N*LN2_HI + LN_FHI_HI
              50     52 40FD  01A2           ADDX    R2, R0                  ; R0 = A + B = ALOG(X)
              50     10    A2 01A6           SUBW2   #1@X_EXP, R0            ; Divide by 2
                           04 01A9           RET
                              01AA
                              01AA      ;
                              01AA      ; Special logic for F close to 1
                              01AA      ;
                              01AA
                              01AA      LN_1_PLUS:
          56  08     50 43FD  01AA           SUBX3   R0, S^#1.0, R6          ; R6 = 1-X
                     55    B5 01AF           TSTW    R5                      ; Determine which way to calculate W
                     12    13 01B1           BEQL    10$
          56  10     56 47FD  01B3           DIVX3   R6, S^#2.0, R6          ; R6 = 2/(1-X)
              56     08 42FD  01B8           SUBX    S^#1.0, R6              ; R6 = (1+X)/(1-X)
              56     55    A2 01BC           SUBW    R5, R6                  ; Scale R6
              56     08 42FD  01BF           SUBX    S^#1.0, R6              ; R6 = W
                     08    11 01C3           BRB     20$
          56  50     56 47FD  01C5      10$: DIVX3   R6, R0, R6              ; R6 = X / (1-X)
              56     10    A0 01CA           ADDW    #1@X_EXP, R6            ; R6 = W = 2*X/(1-X) = (1+X)/(1-X) - 1
     FE3C CF  08     56 55FD  01CD      20$: POLYX   R6, #LOGLEN1, LOGTAB1   ; R0 = Q(W)
              50     56 44FD  01D4           MULX    R6, R0                  ; R0 = W*Q(W)
     52   FE2A CF    6E 45FD  01D8           MULX3   (SP), LN2_LO, R2        ; R2 = N*LN2_LO
              50     52 40FD  01DF           ADDX    R2, R0                  ; R0 = N*LN2_LO + W*Q(W)
              50     56 40FD  01E3           ADDX    R6, R0                  ; R0 = N*LN2_LO + W*Q(W) + W
          6E  FE14 CF 44FD    01E7           MULX    LN2_HI, (SP)            ; (SP) = N*LN2_HI
              50     8E 40FD  01ED           ADDX    (SP)+, R0               ; R0 = ALOG(X)
              50     10    A2 01F1           SUBW2   #1@X_EXP, R0            ; Divide by 2
                           04 01F4           RET
                              01F5      ;
                              01F5      ; X <= 0.0, signal error
                              01F5      ;
        7E    00'8F        9A 01F5      ERROR:  MOVZBL  #MTH$K_INVARGMAT, -(SP) ; condition value
          50  01     0F    79 01F9           ASHQ    #15, #1, R0             ; R0 = result = reserved operand -0.0
                              01FD                                           ; goes to signal mechanism vector
                              01FD                                           ; (CHF$L_MCH_R0/R1) so error handler
                              01FD                                           ; can modify the result.
    00000000'GF   01       FB 01FD           CALLS   #1, G^MTH$$SIGNAL       ; signal error and use real user's PC
```

```
        0204   382                                   ; independent of CALL vs JSB
   04   0204   383          RET                      ; return - R0 restored from
        0205   384                                   ; CHF$L_MCH_R0/R1
        0205   385          .END
```

```
ACMASK           = 000001FC
ERR                00000088 R      01
ERROR              000001F5 R      01
F_EXP            = 00000007
LN2_HI             00000000 R      01
LN2_LO             00000008 R      01
LN_T_PLUS          000001AA R      01
LN_1_PLUS_W        0000014A R      01
LOGLEN1          = 00000008
LOGLEN2          = 00000005
LOGTAB1            00000010 R      01
LOGTAB2            00000058 R      01
MTH$$AB_ALOG_V     ********    X   00
MTH$$AB_G_FHI      ********    X   00
MTH$$SIGNAL        ********    X   00
MTH$GATANH         0000008B RG     01
MTH$K_INVARGMAT    ********    X   00
NEG_EXP            00000139 R      01
X                = 00000004
X_EXP            = 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	00000205 (517.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.85
Command processing	114	00:00:00.71	00:00:04.64
Pass 1	102	00:00:01.37	00:00:04.02
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	81	00:00:00.99	00:00:02.32
Symbol table output	3	00:00:00.00	00:00:00.17
Psect synopsis output	2	00:00:00.01	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	333	00:00:03.16	00:00:12.03

The working set limit was 900 pages.
7026 bytes (14 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 20 non-local and 2 local symbols.
445 source lines were read in Pass 1, producing 11 object records in Pass 2.
3 pages of virtual memory were used to define 2 macros.

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

Macro library name                   Macros defined
------------------                   --------------
_$255$DUA28:[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$:MTHGATANH/OBJ=OBJ$:MTHGATANH MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MS
```

0260 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MTHGCONJG
LIS

MTHGINT
LIS

MTHGMOD
LIS

MTHEXP
LIS

MTHFLOOR
LIS

MTHDTAN
LIS

MTHDTANH
LIS

MTHGEXP
LIS

MTHGFLOOR
LIS

MTHGMIN1
LIS

MTHGCOSH
LIS

MTHGLOG
LIS

MTHGACOS
LIS

MTHGASIN
LIS

MTHGNINT
LIS

MTHGATAN
LIS

MTHGATANH
LIS

MTHGMAX1
LIS