```
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
```

I 12

FILEID**MTHATANH

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

Table of contents

```
0000          .TITLE  MTH$ATANH       ; Single Precision Hyperbolic Arctangent
0000          .IDENT /2-003/          ; File: MTHATANH.MAR  Edit: PDG2003
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$ATANH returns the single precision hyperbolic arctangent of the
0000   ; single 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    ; 1-001 - Algorithm from PL/I math library.
0000    ;
0000    ; Edit History for Version 02 of MTH$ATANH
0000    ;
0000    ; 2-000   Rewrite of PL/I version.      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    ; 2-003 - Repair problem with POLY instruction.  PDG 19-Mar-1982
```

```
          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_F_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    ^X00D0,RL,WL                ; MOVL
          0000        .OPDEF  MOVAX   ^X00DE,AL,WL                ; MOVAL
          0000        .ENDM
          0000
          0000        OPDEF   F, <^X0040>, 0
          0000        .OPDEF  ADDX    ^X00@0+^X0040,RF,MF
          0000        .OPDEF  ADDX3   ^X01@0+^X0040,RF,RF,WF
          0000        .OPDEF  SUBX    ^X02@0+^X0040,RF,MF
          0000        .OPDEF  SUBX3   ^X03@0+^X0040,RF,RF,WF
          0000        .OPDEF  MULX    ^X04@0+^X0040,RF,MF
          0000        .OPDEF  MULX3   ^X05@0+^X0040,RF,RF,WF
          0000        .OPDEF  DIVX    ^X06@0+^X0040,RF,MF
          0000        .OPDEF  DIVX3   ^X07@0+^X0040,RF,RF,WF
          0000        .OPDEF  CVTWX   ^X0D@0+^X0040,RW,WF
          0000        .OPDEF  POLYX   ^X15@0+^X0040,RF,RW,AB
          0000        .OPDEF  MOVX    ^X00D0,RL,WL                ; MOVL
          0000        .OPDEF  MOVAX   ^X00DE,AL,WL                ; MOVAL
          0000
          0000
00000007  0000        F_EXP = 7        ; Bit offset to exponent
00000007  0000        X_EXP = 7        ; 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
        0000003C  0000          ACMASK = ^M<R2,R3,R4,R5>          ; register entry mask and integer
                  0000                                            ; overflow enable
                  0000
        72003CB1  0000  LN2_HI: .LONG   ^X72003CB1                ; (Hi 16 bits of ln2)*2**-7
        BE8E333F  0004  LN2_LO: .LONG   ^XBE8E333F                ; (Low bits of ln2)*2**-7
                  0008
                  0008
                  0008  LOGTAB1:                                  ; Constants for q(z).  Generated using eq.
                  0008                                            ; 6.3.10 of Hart et. al. (sin(2a) = 1/32)
        EABDBF2A  0008          .LONG   ^XEABDBF2A                ; C5 = -.16691108
        0CDD3F4D  000C          .LONG   ^X0CDD3F4D                ; C4 = 0.20024438
        FFF6BF7F  0010          .LONG   ^XFFF6BF7F                ; C3 = -.24999985
        AAA73FAA  0014          .LONG   ^XAAA73FAA                ; C2 = 0.33333322
        0000C000  0018          .LONG   ^X0000C000                ; C1 = -.50000000
                  001C          ;
                  001C          ; Remove this constant, and do another multiply in-line.
                  001C          ;
                  001C  ;;;;;;; .LONG   ^X00000000                ; C0 =  .00000000
        00000004  001C  LOGLEN1 = .-LOGTAB1/4 - 1                 ; no. of floating point entries
                  001C
                  001C
                  001C  LOGTAB2:                                  ; Constants for p(z*z).  Generated using
                  001C                                            ; eq. 6.3.11 of Hart et. al. (sin(2a) =
                  001C                                            ; (b - 1)/(b + 1) where b = 2**(1/7))
        6D943FCD  001C          .LONG   ^X6D943FCD                ; C2 = 0.40122664
        AA91402A  0020          .LONG   ^XAA91402A                ; C1 = 0.66666514
        00004100  0024          .LONG   ^X00004100                ; C0 = 2.00000000
        00000002  0028  LOGLEN2 = .-LOGTAB2/4 - 1
```

```
          0028        .SBTTL  MTH$ATANH       - Single Precision Hyperbolic Arctangent
          0028
          0028  ;++
          0028  ; FUNCTIONAL DESCRIPTION:
          0028  ;
          0028  ; ATANH  - Single precision floating point function
          0028  ;
          0028  ; ATANH(X) is computed using the following approximation technique:
          0028  ;
          0028  ;       If |X| >= 1.0, error.  Otherwise
          0028  ;
          0028  ;       Let (1+X)/(1-X) = f * (2**n), where 1/2 <= f < 1
          0028  ;
          0028  ;       If n is greater than or equal to 1 then
          0028  ;           set N = n - 1 and F1 = 2*f.
          0028  ;       Else
          0028  ;           set N = n and F = f.
          0028  ;
          0028  ;       If |F - 1| < 2**-5 then
          0028  ;           2*atanh(X) =  N*ln(2) + W + W*P(W),
          0028  ;               where W = ((1+F)/(1-F))*2**N - 1,
          0028  ;               and P is a polynomial of degree F=5,D=9.
          0028  ;       Else
          0028  ;           Obtain FHI (roughly equal to F) from table lookup.
          0028  ;           2*atanh(X) = ln((1+X)/(1-X)) = N*ln(2) + ln(FHI) + Z*Q(Z*Z),
          0028  ;               where Q is a polynomial of degree F=2,D=5,
          0028  ;               where Z = (F - FHI)/(F + FHI)
          0028  ;               where F = (2**-N)*(1+X)/(1-X)
          0028  ;           Z is computed by:
          0028  ;               Z = (X-D)/(1-X*D)
          0028  ;               where Y = FHI*2**N
          0028  ;               where D = (Y-1)/(Y+1)
          0028  ;           Note that Z may be computed in a variety of ways:
          0028  ;               Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
          0028  ;               Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
          0028  ;               Z = [1 - Y + X + X*Y]/[1 + Y + X - X*Y]
          0028  ;               Z = [(1-Y) + X*(1+Y)]/[(1+Y) + X*(1-Y)]
          0028  ;
          0028  ;       NOTE:   The quantities ln(A=FHI) and ln2 are used in the above
          0028  ;               equations in two parts - a high part (containing the
          0028  ;               high order bits) and a low part (containing the low
          0028  ;               order bits.  In the code the high and low parts of the
          0028  ;               constants are indicated by a _HI and _LO suffix respec-
          0028  ;               tively.  The values were chosen such that N*LN2_HI +
          0028  ;               LN_FHI_HI is exactly representable.
          0028  ;
          0028  ; CALLING SEQUENCE:
          0028  ;
          0028  ;       atanh.wf.v = MTH$ATANH(x.rf.r)
          0028  ;
          0028  ; INPUT PARAMETERS:
          0028
00000004  0028        x = 4                           ; Contents of x is the argument
          0028
          0028  ; IMPLICIT INPUTS:      none
          0028  ;
          0028  ; OUTPUT PARAMETERS:
```

```
                              0028  ;
                              0028  ;       VALUE:  Single precision hyperbolic arctangent of the argument
                              0028  ;
                              0028  ; IMPLICIT OUTPUTS:      none
                              0028  ;
                              0028  ; COMPLETION CODES:      none
                              0028  ;
                              0028  ; SIDE EFFECTS:
                              0028  ;
                              0028  ; Signals: MTH$K_INVARGMAT if |X| >= 1.0 with reserved operand in R0 (copied to
                              0028  ; the signal mechanism vector CHF$L_MCH_R0/R1 by LIB$SIGNAL).
                              0028  ; Associated message is: "Floating overflow in math library". Result is
                              0028  ; reserved operand -0.0 unless a user supplied (or any) error handler changes
                              0028  ; CHF$L_MCH_R0/R1.
                              0028  ;
                              0028  ; NOTE: This procedure disables floating point underflow and integer
                              0028  ; overflow, causes no floating overflow or other arithmetic traps, and
                              0028  ; preserves enables across the call.
                              0028  ;
                              0028  ; Note: This routine is written to avoid causing any integer overflows,
                              0028  ; floating overflows, or floating underflows or divide by 0 conditions,
                              0028  ; whether enabled or not.
                              0028  ;
                              0028  ;---
                              0028
              0134  31        0028  ERR:    BRW     ERROR
                              002B
                    003C      002B          .ENTRY  MTH$ATANH, ACMASK       ; standard call-by-reference entry
                              002D                                          ; disable DV (and FU), enable IV
     50     04 BC   D0        002D          MOVX    @X(AP), R0              ; R0 = arg
                              0031
  52    08    50    43        0031          SUBF3   R0, S^#1.0, R2          ; R2 = 1-X
              F1    15        0035          BLEQ    ERR                     ; ATANH(X) is not defined for X>=1
  54    08    50    41        0037          ADDF3   R0, S^#1.0, R4          ; R4 = 1+X
              EB    15        003B          BLEQ    ERR                     ; ATANH(X) is not defined for X<=-1
        54    52    46        003D          DIVF2   R2, R4                  ; R4 = approximation to (1+X)/(1-X)
 55    00000000'GF  9E        0040          MOVAB   G^MTH$$AB_ALOG_V, R5
        55    65    C0        0047          ADDL2   (R5), R5                ; R5 = address of ALOG table
53   54  007F 8F    AB        004A          BICW3   #1@F_EXP-1, R4, R3      ; R3 = Biased exponent
  53     4000 8F    A2        0050          SUBW    #^X4000, R3             ; R3 = Unbiased exponent
              63    15        0055          BLEQ    NEG_EXP                 ; Branch to processing for n=<0
                              0057
  53     0080 8F    A2        0057          SUBW    #1@F_EXP, R3            ; Exponent is positive, R3 = N = n - 1
        54    53    A2        005C          SUBW    R3, R4                  ; R4 = F = 2f
        54    54    9A        005F          MOVZBL  R4, R4                  ; R4 = index into ALOG table
              00000000        0062          .IF     NE, F_EXP-X_EXP
                              0062          DIVW2   #1@<F_EXP-X_EXP>, R3    ; Shift R3 to scale X-floating
                              0062          .ENDC
        7E    53    4D        0062          CVTWX   R3, -(SP)               ; Push N onto the stack
    55      6544    98        0065          CVTBL   (R5)[R4], R5            ; R5 = offset into FHI tables
              5C    19        0069          BLSS    LN_1_PLUS_W             ; Branch to handle F close to 1
55    00000000'GF45 DE        006B          MOVAX   G^MTH$$AB_F_FHI[R5], R5 ; R5 = Address of FHI
        54    85    D0        0073          MOVX    (R5)+, R4               ; R4 = FHI
                              0076  ;
                              0076  ; Compute Z = (F - FHI)/(F + FHI)
                              0076  ;                 Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
                              0076  ;                 Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
```

```
                         0076  ;        where Y = FHI*2**N, roughly equal to (1+X)/(1-X)
                         0076  ;
         54    53    A0  0076          ADDW     R3, R4                    ; R4 = FHI * 2**N = SFHI
    52   08    54    43  0079          SUBX3    R4, S^#1.0, R2            ; R2 = 1 - SFHI
         54    08    40  007D          ADDX     S^#1.0, R4                ; R4 = 1 + SFHI
         52    54    46  0080          DIVX     R4, R2                    ; R2 = (1-SFHI)/(1+SFHI) = D
    54   52    50    41  0083          ADDX3    R0, R2, R4                ; R4 = D + X
         52    50    44  0087          MULX     R0, R2                    ; R2 = D * X
         52    08    40  008A          ADDX     S^#1.0, R2                ; R2 = 1 + D*X
         54    52    46  008D          DIVX     R2, R4                    ; R4 = (D+X)/(1+D*X) = Z
                         0090  ;
                         0090  ; Compute Z**2, P(Z**2) and Z*P(Z**2)
                         0090  ;
    50   54    54    45  0090          MULX3    R4, R4, R0                ; R0 = Z**2
 83 AF   02    50    55  0094          POLYX    R0, #LOGLEN2, LOGTAB2     ; R0 = P(Z**2)
         50    54    44  0099          MULX     R4, R0                    ; R0 = Z*P(Z**2)
                         009C  ;
                         009C  ; Compute B = N*LN2_LO + LN_FHI_LO + Z*P(Z*Z)
                         009C  ;
  52   FF63 CF   6E  45  009C          MULX3    (SP), LN2_LO, R2          ; R2 = N*LN2_LO
         52    85    40  00A2          ADDX     (R5)+, R2                 ; R2 = N*LN2_LO + LN_FHI_LO
         50    52    40  00A5          ADDX     R2, R0                    ; R0 = B
                         00A8  ;
                         00A8  ; Compute A = N*LN2_HI + LN_FHI_HI and ALOG(X)
                         00A8  ;
  52   FF53 CF   8E  45  00A8          MULX3    (SP)+, LN2_HI, R2         ; R2 = N*LN2_HI
         52    65    40  00AE          ADDX     (R5), R2                  ; R2 = A = N*LN2_HI + LN_FHI_HI
         50    52    40  00B1          ADDX     R2, R0                    ; R0 = A + B = ALOG(X)
    50   0080 8F     A2  00B4          SUBW2    #1@X_EXP, R0              ; Divide by 2
                     04  00B9          RET
                         00BA
                         00BA  NEG_EXP:
         54    53    A2  00BA          SUBW     R3, R4                    ; R4 = F = 2f
         54    54    9A  00BD          MOVZBL   R4, R4                    ; R4 = index into ALOG table
            00000000     00C0          .IF      NE, F_EXP-X_EXP
                         00C0          DIVW2    #1@<F_EXP-X_EXP>, R3      ; Shift R3 to scale X-floating
                         00C0          .ENDC
         7E    53    4D  00C0          CVTWX    R3, -(SP)                 ; Push N onto the stack
      55   6544      98  00C3          CVTBL    (R5)[R4], R5              ; R5 = offset into FHI tables
                         00C7  LN_1_PLUS_W:
               50    19  00C7          BLSS     LN_1_PLUS                 ; Branch to handle F close to 1
 55   00000000'GF45  DE  00C9          MOVAX    G^MTH$$AB_F_FHI[R5], R5   ; R5 = Address of FHI
         54    65    D0  00D1          MOVX     (R5), R4                  ; R4 = FHI
                         00D4  ;
                         00D4  ; Compute Z = (F - FHI)/(F + FHI)
                         00D4  ;               Z = [(1+X) - Y*(1-X)]/[(1+X) + Y*(1-X)]
                         00D4  ;               Z = [1 + X - Y + X*Y]/[1 + X + Y - X*Y]
                         00D4  ;        where Y = FHI*2**N, roughly equal to (1+X)/(1-X)
                         00D4  ;
         54    53    A0  00D4          ADDW     R3, R4                    ; R4 = FHI * 2**N = SFHI
    52   08    54    43  00D7          SUBX3    R4, S^#1.0, R2            ; R2 = 1 - SFHI
         54    08    40  00DB          ADDX     S^#1.0, R4                ; R4 = 1 + SFHI
         52    54    46  00DE          DIVX     R4, R2                    ; R2 = (1-SFHI)/(1+SFHI) = D
    54   52    50    41  00E1          ADDX3    R0, R2, R4                ; R4 = D + X
         52    50    44  00E5          MULX     R0, R2                    ; R2 = D * X
         52    08    40  00E8          ADDX     S^#1.0, R2                ; R2 = 1 + D*X
         54    52    46  00EB          DIVX     R2, R4                    ; R4 = (D+X)/(1+D*X) = Z
```

```
                                00EE        ;
                                00EE        ; Compute Z**2, P(Z**2) and Z*P(Z**2)
                                00EE        ;
          50    54    54    45  00EE                MULX3   R4, R4, R0              ; R0 = Z**2
     FF24 CF    02    50    55  00F2                POLYX   R0, #LOGLEN2, LOGTAB2   ; R0 = P(Z**2)
                50    54    44  00F8                MULX    R4, R0                  ; R0 = Z*P(Z**2)
                                00FB        ;
                                00FB        ; Compute B = N*LN2_LO + LN_FHI_LO + Z*P(Z*Z)
                                00FB        ;
     52    FF04 CF    6E    45  00FB                MULX3   (SP), LN2_LO, R2        ; R2 = N*LN2_LO
                52    75    40  0101                ADDX    -(R5), R2               ; R2 = N*LN2_LO + LN_FHI_LO
                50    52    40  0104                ADDX    R2, R0                  ; R0 = B
                                0107        ;
                                0107        ; Compute A = N*LN2_HI + LN_FHI_HI and ALOG(X)
                                0107        ;
     52    FEF4 CF    8E    45  0107                MULX3   (SP)+, LN2_HI, R2       ; R2 = N*LN2_HI
                52    75    42  010D                SUBX    -(R5), R2               ; R2 = A = N*LN2_HI + LN_FHI_HI
                50    52    40  0110                ADDX    R2, R0                  ; R0 = A + B = ALOG(X)
          50    0080 8F     A2  0113                SUBW2   #1@X_EXP, R0            ; Divide by 2
                            04  0118                RET
                                0119
                                0119        ;
                                0119        ; Special logic for F close to 1
                                0119        ;
                                0119
                                0119        LN_1_PLUS:
          54    08    50    43  0119                SUBX3   R0, S^#1.0, R4          ; R4 = 1-X
                      53    B5  011D                TSTW    R3                      ; Determine which way to calculate W
                      0F    13  011F                BEQL    10$
          54    10    54    47  0121                DIVX3   R4, S^#2.0, R4          ; R4 = 2/(1-X)
                54    08    42  0125                SUBX    S^#1.0, R4              ; R4 = (1+X)/(1-X)
                54    53    A2  0128                SUBW    R3, R4                  ; Scale R4
                54    08    42  012B                SUBX    S^#1.0, R4              ; R4 = W
                      09    11  012E                BRB     20$
          54    50    54    47  0130        10$:    DIVX3   R4, R0, R4              ; R4 = X / (1-X)
          54    0080 8F     A0  0134                ADDW    #1@X_EXP, R4            ; R4 = W = 2*X/(1-X) = (1+X)/(1-X) - 1
     FEC9 CF    04    54    55  0139        20$:    POLYX   R4, #LOGLEN1, LOGTAB1   ; R0 = Q(W)
                50    54    44  013F                MULX    R4, R0                  ; Finish computing Q(W)
                50    54    44  0142                MULX    R4, R0                  ; R0 = W*Q(W)
     52    FEBA CF    6E    45  0145                MULX3   (SP), LN2_LO, R2        ; R2 = N*LN2_LO
                50    52    40  014B                ADDX    R2, R0                  ; R0 = N*LN2_LO + W*Q(W)
                50    54    40  014E                ADDX    R4, R0                  ; R0 = N*LN2_LO + W*Q(W) + W
          6E    FEAB CF     44  0151                MULX    LN2_HI, (SP)            ; (SP) = N*LN2_HI
                50    8E    40  0156                ADDX    (SP)+, R0               ; R0 = ALOG(X)
          50    0080 8F     A2  0159                SUBW2   #1@X_EXP, R0            ; Divide by 2
                            04  015E                RET
                                015F        ;
                                015F        ; X <= 0.0, signal error
                                015F        ;
          7E    00'8F       9A  015F        ERROR:  MOVZBL  #MTH$K_INVARGMAT, -(SP) ; condition value
          50    01    0F    78  0163                ASHL    #15, #1, R0             ; R0 = result = reserved operand -0.0
                                0167                                                ; goes to signal mechanism vector
                                0167                                                ; (CHF$L_MCH_R0/R1) so error handler
                                0167                                                ; can modify the result.
      00000000'GF   01      FB  0167                CALLS   #1, G^MTH$$SIGNAL       ; signal error and use real user's PC
                                016E                                                ; independent of CALL vs JSB
                            04  016E                RET                             ; return - R0 restored from
```

```
016F    373                              ; CHF$L_MCH_R0/R1
016F    374         .END
```

```
ACMASK           = 0000003C
ERR                00000028 R      01
ERROR              0000015F R      01
F_EXP            = 00000007
LN2_HI             00000000 R      01
LN2_LO             00000004 R      01
LN_1_PLUS          00000119 R      01
LN_1_PLUS_W        000000C7 R      01
LOGLEN1          = 00000004
LOGLEN2          = 00000002
LOGTAB1            00000008 R      01
LOGTAB2            0000001C R      01
MTH$$AB_ALOG_V     ********     X  00
MTH$$AB_F_FHI      ********     X  00
MTH$$SIGNAL        ********     X  00
MTH$ATANH          0000002B RG     01
MTH$K_INVARGMAT    ********     X  00
NEG_EXP            000000BA R      01
X                = 00000004
X_EXP            = 00000007
```

```
                                    +----------------+
                                    ! 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                   0000016F  (  367.)  01 (  1.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD   NOWRT NOVEC LONG

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

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   33     00:00:00.08    00:00:00.72
Command processing              126     00:00:00.67    00:00:04.14
Pass 1                           94     00:00:01.28    00:00:04.37
Symbol table sort                 0     00:00:00.01    00:00:00.01
Pass 2                           85     00:00:00.92    00:00:03.66
Symbol table output               3     00:00:00.03    00:00:00.03
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            345     00:00:03.02    00:00:12.98

The working set limit was 900 pages.
6708 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.
434 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$:MTHATANH/OBJ=OBJ$:MTHATANH MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC
```

0257 AH-BT13A-SE
VAX/VMS V4.0

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