F 13

```
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
```

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

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
```

```
0000            .TITLE  MTH$HATAN       ; H Floating Point Arc Tangent Functions
0000                                    ; (HATAN,HATAN2,HATAND,HATAND2)
0000            .IDENT /2-006/          ; File: MTHHATAN.MAR  EDIT:  RNH2006
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$HATAN  is  a function which returns the H floating point arctangent
0000    ; value (in radians) of its H floating point argument.  MTH$HATAN2 is two
0000    ; argument H floating arctangent.  The call is standard call-by-reference.
0000    ; MTH$HATAN_R8 is a special routine which is the same as MTH$HATAN except
0000    ; that a faster non-standard JSB call is used with the argument in  R0-R3
0000    ; and no registers are saved.
0000    ;
0000    ; MTH$HATAND  is  a  function which returns the H floating point arctangent
0000    ; value (in degrees) of its H floating point argument.  MTH$HATAND2 is  two
0000    ; argument  H floating arctangent.  The call is standard call-by-reference.
0000    ; MTH$HATAND_R8 is a special routine which is the same as MTH$HATAND except
0000    ; that  a  faster  non-standard JSB call is used with the argument in R0-R3
0000    ; and no registers are saved.
0000    ;
0000    ;--
0000    ;
0000    ; VERSION: 1
0000    ;
0000    ; HISTORY:
0000    ; AUTHOR:
0000    ;       John A. Wheeler, 03-Oct-1979: Version 1
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;
```

```
0000 ; VERSION: 2
0000 ;
0000 ; HISTORY:
0000 ; AUTHOR:
0000 ;       Bob Hanek, 10-Jun-1981: Version 2
0000 ;
0000 ; MODIFIED BY:
0000 ;
0000 ;
0000 ;
```

```
0000
0000
0000      ; Edit history for Version 1 of MTH$HATAN
0000      ;
0000      ; 1-001 - Adapted from MTH$GATAN version 1-001.  JAW 03-Oct-1979.
0000      ; 1-002 - Added degree entry points. RNH 15-MAR-1981
0000      ;
0000      ;
0000      ; Edit history for Version 1 of MTH$HATAN
0000      ;
0000      ; 2-002 - Use G^ addressing for externals.  SBL 24-Aug-1981
0000      ; 2-003 - Changed MTH$$AB_ATAN to MTH$$AB_ATAN_V.  RNH 29-Sep-81
0000      ; 2-004 - Change MTH$HATAND2 entry to MTH$HATAN2D in order to conform
0000      ;         with original specification.  RNH 05-Oct-81
0000      ; 2-005 - Changed TSTB to TSTH in MTH$HATAND.  RNH 05-Oct-81
0000      ; 2-006 - Un-did edit 2-004 to comform with PL/1.
0000      ;       - Modified small argument logic to avoid a microcode bug in the
0000      ;         FPA. RNH 18-Dec-81
```

```
                                         0000        .SBTTL  DECLARATIONS           ; Declarative Part of Module
                                         0000
                                         0000 ;
                                         0000 ; INCLUDE FILES:       MTHJACKET.MAR, MTHATAN.MAR
                                         0000 ;
                                         0000 ; EXTERNAL SYMBOLS:
                                         0000 ;
                                         0000        .DSABL  GBL
                                         0000        .EXTRN  MTH$K_INVARGMAT
                                         0000        .EXTRN  MTH$$SIGNAL            ; Signal SEVERE error
                                         0000        .EXTRN  MTH$$AB_ATAN_V         ; Gobal table used by all Arctangent
                                         0000                                       ;    routines.  Part of MTHATAN.MAR
                                         0000 ;
                                         0000
                                         0000 ; EQUATED SYMBOLS:
                                         0000 ;
                               000041FC  0000        ACMASK = ^M<IV, R2, R3, R4, R5, R6, R7, R8>     ; .ENTRY register mask,
                                         0000                                                        ; int ovf enabled
                                         0000
                                         0000 ;
                                         0000 ; MACROS:       none
                                         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 ; EXTERNALS:
                                         0000 ;
                                         0000        .EXTRN  MTH$$SIGNAL            ; Signal a severe error
                                         0000        .EXTRN  MTH$K_INVARGMAT        ; Invalid argument to math library
                                         0000        .DSABL  GBL                    ; No other externals allowed
                                         0000 ;
                                         0000 ; CONSTANTS:
                                         0000 ;
00000000 00000000 00000000 0000C001      0000 H_M1.0: .H_FLOATING     -1.0
                                         0010
```

```
                                      0010  ;
                                      0010  ;    ************************* Constants for HATAN ***************************
                                      0010  ;
                                      0010  ;
                                      0010  ;    Each  entry of the HATAN_TABLE contains the the values of XHI, HATAN_XHI_LO
                                      0010  ;    and HATAN_XHI_HI respectively.  The table is indexed by a pointer  obtained
                                      0010  ;    from the MTH$$AB_ATAN_V table. The MTH$$AB_ATAN_V table is common to all of the
                                      0010  ;    arctangent  routines  and is included as part of the MTHATAN module.  NOTE:
                                      0010  ;    For performance reasons it is important to have  the  HATAN_TABLE  longword
                                      0010  ;    aligned.
                                      0010  ;
                                      0010  ;
                                      0010          .ALIGN  LONG
                                      0010
                                      0010  HATAN_TABLE:
                                      0010   ; Entry  0
00000000 00000000 0000FC00 AFF03FFD   0010          .OCTA   ^X0000000000000000000FC00AFF03FFD ; .105454429984092712402343750000
3F80047B 6BE701F0 8954A2CE 08383F8B   0020          .OCTA   ^X3F80047B6BE701F08954A2CE08383F8B ; .310590568209781578539862675675
1DE4A44B 1AD2ED08 F566CD64 AE593FFD   0030          .OCTA   ^X1DE4A44B1AD2ED08F566CD64AE593FFD ; .105066110917812355541860264623
                                      0040   ; Entry  1
00000000 00000000 0000E000 07F63FFE   0040          .OCTA   ^X000000000000000000000E00007F63FFE ; .128888845443725585937500000000
9DC0001F 3C7BEDB0 5D58B33C C4F9BF8B   0050          .OCTA   ^X9DC0001F3C7BEDB05D58B33CC4F9BF8B ;-.532470013383326145019230697183
D38AF577 F4A3F5FB 79B85E6F 06843FFE   0060          .OCTA   ^XD38AF577F4A3F5FB79B85E6F06843FFE ; .128182161118470788030782153769
                                      0070   ; Entry  2
00000000 00000000 00007C00 3FEC3FFE   0070          .OCTA   ^X0000000000000000000007C003FEC3FFE ; .156212776899337768554687500000
1000BAA5 2EA43F36 6EF48C38 171BBF86   0080          .OCTA   ^X1000BAA52EA43F366EF48C38171BBF86 ;-.102527942775025756423409531871
215F05DE B08D24D7 266DE196 3D5B3FFE   0090          .OCTA   ^X215F05DEB08D24D7266DE1963D5B3FFE ; .154960405726163383340293800028
                                      00A0   ; Entry  3
00000000 00000000 0000D800 8FC93FFE   00A0          .OCTA   ^X00000000000000000000D8008FC93FFE ; .195209205150604248046875000000
F96044D9 88E42802 BE817206 4B3EBF8C   00B0          .OCTA   ^XF96044D988E42802BE8172064B3EBF8C ;-.778750730815030023823483290232
8D85AE96 15FCE5FF B59EC355 8AD23FFE   00C0          .OCTA   ^X8D85AE9615FCE5FFB59EC3558AD23FFE ; .192784811070580496671823196569
                                      00D0   ; Entry  4
00000000 00000000 0000A200 FF873FFE   00D0          .OCTA   ^X00000000000000000000A200FF873FFE ; .249770417809486389160156250000
3E603120 E252BDEB 9C3BFB18 5A8A3F8C   00E0          .OCTA   ^X3E603120E252BDEB9C3BFB185A8A3F8C ; .814718428248051280119869935834
258E6D50 B04E4DA6 678D1498 F5463FFE   00F0          .OCTA   ^X258E6D50B04E4DA6678D1498F5463FFE ; .244762574101463538467234953650
                                      0100   ; Entry  5
00000000 00000000 00002E00 3FB73FFF   0100          .OCTA   ^X000000000000000000002E003FB73FFF ; .312222212553024291992187500000
F740A18A 31419CDF 169664BE 520EBF8D   0110          .OCTA   ^XF740A18A31419CDF169664BE520EBF8D ;-.158953272962318297496813319617
A5952079 CA576800 315D1AA3 35E53FFF   0120          .OCTA   ^XA5952079CA576800315D1AA335E53FFF ; .302631775103092165457195172037
                                      0130   ; Entry  6
00000000 00000000 00001C00 8F3D3FFF   0130          .OCTA   ^X000000000000000000001C008F3D3FFF ; .389881551265716552734375000000
2FD07DB2 D14B7DE7 945D0A51 B4E9BF8D   0140          .OCTA   ^X2FD07DB2D14B7DE7945D0A51B4E9BF8D ;-.205434242228330560627520199599
64221FA2 13F6C3B2 F1A5E2DE 7CAC3FFF   0150          .OCTA   ^X64221FA213F6C3B2F1A5E2DE7CAC3FFF ; .371753258569162316536708564816
                                      0160   ; Entry  7
00000000 00000000 00006C00 FE673FFF   0160          .OCTA   ^X000000000000000000006C00FE673FFF ; .498441398143768310546875000000
FFD06E7E 2DC38FDC 14094F56 B263BF8D   0170          .OCTA   ^XFFD06E7E2DC38FDC14094F56B263BF8D ;-.204248222794771760669725277858
178F75CA 1E820D4E 36605F61 D97F3FFF   0180          .OCTA   ^X178F75CA1E820D4E36605F61D97F3FFF ; .462399950321554388897430653076
                                      0190   ; Entry  8
00000000 00000000 0000D600 4DF14000   0190          .OCTA   ^X00000000000000000000D6004DF14000 ; .652235686779022216796875000000
3D40054F B21265C9 4ACA21A1 4AF23F8C   01A0          .OCTA   ^X3D40054FB21265C94ACA21A14AF23F8C ; .778049896968520132497663221063
FA28AA48 20BA6844 B96B7173 27E84000   01B0          .OCTA   ^XFA28AA4820BA6844B96B717327E84000 ; .577945275665760894071884596911
                                      01C0   ; Entry  9
00000000 00000000 00002400 BC0E4000   01C0          .OCTA   ^X000000000000000000002400BC0E4000 ; .867295384407043457031250000000
6FA01DA8 9E2223B7 702B5C96 BE76BF8D   01D0          .OCTA   ^X6FA01DA89E2223B7702B5C96BE76BF8D ;-.209925588241432789066125702518
0D41BDF1 24DF7738 95CC5766 6DCC4000   01E0          .OCTA   ^X0D41BDF124DF773895CC57666DCC4000 ; .714449626228906115167021171246
                                      01F0   ; Entry 10
00000000 00000000 0000B000 2B974001   01F0          .OCTA   ^X00000000000000000000B0002B974001 ; .117028331756591796875000000000
```

```
1A209628 58DA5159 6AB88059 7317BF8E  0200          .OCTA   ^X1A20962858DA51596AB880597317BF8E ;-.34897305370243262708530700956O
968C64F7 D2B8A513 44CCC404 BA364000  0210          .OCTA   ^X968C64F7D2B8A51344CCC404BA364000 ; .86369907905682311769930752422 2
                                     0220   ; Entry 11
00000000 00000000 0000D600 A51B4001  0220          .OCTA   ^X000000000000000000000D600A51B4001 ; .16449559926986694335937500000 0
000005E5 35CBD647 B26716D8 5EEDBF8F  0230          .OCTA   ^X000005E535CBD647B26716D85EEDBF8F ;-.66001869312923491507163783448 8
A226B1AA 552CA0AC 12038186 064A4001  0240          .OCTA   ^XA226B1AA552CA0AC12038186064A4001 ; .10245743706054910675703287370 5
                                     0250   ; Entry 12
00000000 00000000 00000A00 49104002  0250          .OCTA   ^X00000000000000000000A0049104002 ; .25708019733428955078125000000 0
00009BF3 551AE950 6AFE1099 A62E3F8F  0260          .OCTA   ^X00009BF3551AE9506AFE1099A62E3F8F ; .79403221667729798157510089905 5
D3691BCA CD087ACB 888B94B3 33274001  0270          .OCTA   ^XD3691BCACD087ACB888B94B333274001 ; .11998227060617362826358168340 8
                                     0280   ; Entry 13
00000000 00000000 00001C00 56FA4003  0280          .OCTA   ^X0000000000000000000001C0056FA4003 ; .53590154647827148437500000000 0
000007B2 E2865FE2 3A6884AE 03E13F8F  0290          .OCTA   ^X000007B2E2865FE23A6884AE03E13F8F ; .48878170565746800023783673028 5
8C47E8A7 BCA66957 6FBFA464 62E54001  02A0          .OCTA   ^X8C47E8A7BCA669576FBFA46462E54001 ; .13863165612417539715828567037 9
                                     02B0
                                     02B0   ;
                                     02B0   ; Tables to be used in POLYH for computing HATAN:  HATANTAB1 is obtained
                                     02B0   ; from Hart et. al. (Eq. 6.5.16, p = 3/32).  HATANTAB2 is the same as HATANTAB1
                                     02B0   ; except that C0 is set to 0
                                     02B0   ;
                                     02B0
                                     02B0   HATANTAB1:
7C347736 74B4E0AC DB3F82FB 09193FFC  02B0          .OCTA   ^X7C34773674B4E0ACDB3F82FB09193FFC      ; C14 =  0.32360797717821285
A47394F6 F8BAC976 B32900C9 2EE1BFFC  02C0          .OCTA   ^XA47394F6F8BAC976B32900C92EE1BFFC      ; C13 = -0.36972524203155308
132BA1B3 2A90F4D2 BE7A9C35 47AB3FFC  02D0          .OCTA   ^X132BA1B32A90F4D2BE7A9C3547AB3FFC      ; C12 =  0.39998822305235543
81D9AD10 1478D7B0 CEB67DDF 642CBFFC  02E0          .OCTA   ^X81D9AD101478D7B0CEB67DDF642CBFFC      ; C11 = -0.43478246544018311
DC990115 16D26D09 4BC46175 86183FFC  02F0          .OCTA   ^XDC99011516D26D094BC4617586183FFC      ; C10 =  0.47619047496816827
80B957F8 3D2C6893 00886BCA AF28BFFC  0300          .OCTA   ^X80B957F83D2C689300886BCAAF28BFFC      ; C09 = -0.52631578946617675
364FC3E7 1A5BAC61 E1C3E1E1 E1E13FFC  0310          .OCTA   ^X364FC3E71A5BAC61E1C3E1E1E1E13FFC      ; C08 =  0.58823529411761352
88932810 EB160484 11111111 1111BFFD  0320          .OCTA   ^X88932810EB160484111111111111BFFD      ; C07 = -0.66666666666666655
26F87B8B 36633B0C 13B1B13B 3B133FFD  0330          .OCTA   ^X26F87B8B36633B0C13B1B13B3B133FFD      ; C06 =  0.76923076923076923
B415D5C5 42B15D17 D1741745 745DBFFD  0340          .OCTA   ^XB415D5C542B15D17D1741745745DBFFD      ; C05 = -0.90909090909090909
CD4F3BAB 71C6C71C 1C7171C7 C71C3FFD  0350          .OCTA   ^XCD4F3BAB71C6C71C1C7171C7C71C3FFD      ; C04 =  0.11111111111111111
30309235 49242492 92494924 2492BFFE  0360          .OCTA   ^X30309235492424929249492424 92BFFE     ; C03 = -0.14285714285714285
97A79999 99999999 99999999 99993FFE  0370          .OCTA   ^X97A79999999999999999999999993FFE      ; C02 =  0.19999999999999999
554E5555 55555555 55555555 5555BFFF  0380          .OCTA   ^X554E555555555555555555555555BFFF      ; C01 = -0.33333333333333333
00000000 00000000 00000000 00004001  0390          .OCTA   ^X00000000000000000000000000004001      ; C00 =  0.10000000000000000
                            0000000F  03A0   HATANLEN1 = .- HATANTAB1/16
                                     03A0
                                     03A0   HATANTAB2:
7C347736 74B4E0AC DB3F82FB 09193FFC  03A0          .OCTA   ^X7C34773674B4E0ACDB3F82FB09193FFC      ; C14 =  0.32360797717821285
A47394F6 F8BAC976 B32900C9 2EE1BFFC  03B0          .OCTA   ^XA47394F6F8BAC976B32900C92EE1BFFC      ; C13 = -0.36972524203155308
132BA1B3 2A90F4D2 BE7A9C35 47AB3FFC  03C0          .OCTA   ^X132BA1B32A90F4D2BE7A9C3547AB3FFC      ; C12 =  0.39998822305235543
81D9AD10 1478D7B0 CEB67DDF 642CBFFC  03D0          .OCTA   ^X81D9AD101478D7B0CEB67DDF642CBFFC      ; C11 = -0.43478246544018311
DC990115 16D26D09 4BC46175 86183FFC  03E0          .OCTA   ^XDC99011516D26D094BC4617586183FFC      ; C10 =  0.47619047496816827
80B957F8 3D2C6893 00886BCA AF28BFFC  03F0          .OCTA   ^X80B957F83D2C689300886BCAAF28BFFC      ; C09 = -0.52631578946617675
364FC3E7 1A5BAC61 E1C3E1E1 E1E13FFC  0400          .OCTA   ^X364FC3E71A5BAC61E1C3E1E1E1E13FFC      ; C08 =  0.58823529411761352
88932810 EB160484 11111111 1111BFFD  0410          .OCTA   ^X88932810EB160484111111111111BFFD      ; C07 = -0.66666666666666655
26F87B8B 36633B0C 13B1B13B 3B133FFD  0420          .OCTA   ^X26F87B8B36633B0C13B1B13B3B133FFD      ; C06 =  0.76923076923076923
B415D5C5 42B15D17 D1741745 745DBFFD  0430          .OCTA   ^XB415D5C542B15D17D1741745745DBFFD      ; C05 = -0.90909090909090909
CD4F3BAB 71C6C71C 1C7171C7 C71C3FFD  0440          .OCTA   ^XCD4F3BAB71C6C71C1C7171C7C71C3FFD      ; C04 =  0.11111111111111111
30309235 49242492 92494924 2492BFFE  0450          .OCTA   ^X30309235492424929249492424 92BFFE     ; C03 = -0.14285714285714285
97A79999 99999999 99999999 99993FFE  0460          .OCTA   ^X97A79999999999999999999999993FFE      ; C02 =  0.19999999999999999
554E5555 55555555 55555555 5555BFFF  0470          .OCTA   ^X554E555555555555555555555555BFFF      ; C01 = -0.33333333333333333
00000000 00000000 00000000 00000000  0480          .OCTA   ^X00000000000000000000000000000000      ; C00 =  0.00000000000000000
                            0000000F  0490   HATANLEN2 = .- HATANTAB2/16
                                     0490
```

```
                                      0490   242 H_PI:
01B8C517 898C8469 42D1B544 921F4002   0490   243         .OCTA   ^X01B8C517898C846942D1B544921F4002      ; pi
                                      04A0   244 H_PI_OVER_2:
01B8C517 898C8469 42D1B544 921F4001   04A0   245         .OCTA   ^X01B8C517898C846942D1B544921F4001      ; pi/2
                                      04B0   246 H_MPI_OVER_2:
01B8C517 898C8469 42D1B544 921FC001   04B0   247         .OCTA   ^X01B8C517898C846942D1B544921FC001      ; -pi/2
                                      04C0   248 H_PI_OVER_2_HI:
01B8C517 898C8469 42D1B544 921F4001   04C0   249         .OCTA   ^X01B8C517898C846942D1B544921F4001      ; High order bits of pi/2
                                      04D0   250 H_PI_OVER_2_LO:
000020BA C740A67C E0889024 CD123F8E   04D0   251         .OCTA   ^X000020BAC740A67CE0889024CD123F8E      ; Low order bits of pi/2
                                      04E0   252
```

```
                                     04E0  ;
                                     04E0  ; ************************* Constants for HATAND ******************************
                                     04E0  ;
                                     04E0  ; Each entry of the HATAND_TABLE contains the the values of XHI, HATAND_XHI_LO
                                     04E0  ; and HATAND_XHI_HI respectively.  The table is indexed by a pointer  obtained
                                     04E0  ; from  the MTH$$AB_ATAN_V table. The MTH$$AB_ATAN_V table is common to all of  the
                                     04E0  ; arctangent routines and is included as part of the MTHATAN module. NOTE: For
                                     04E0  ; performance reasons it is important to have the HATAN_TABLE longword aligned.
                                     04E0  ;
                                     04E0  ;
                                     04E0
                                     04E0  HATAND_TABLE:
                                     04E0   ; Entry  0
00000000 00000000 0000FC00 AFF03FFD  04E0           .OCTA   ^X000000000000000000000FC00AFF03FFD ; .1054544299840927124023437500000
00004ECA 4446B3F6 A4918101 0F57BF8F  04F0           .OCTA   ^X00004ECA4446B3F6A49181010F57BF8F ;-.5103372278724120098062154746l6
04DF0735 3535C7CC B1EE22CF 81454003  0500           .OCTA   ^X04DF07353535C7CCB1EE22CF81454003 ; .6019844725444027921806115863O6
                                     0510   ; Entry  1
00000000 00000000 0000E000 07F63FFE  0510           .OCTA   ^X0000000000000000000E00007F63FFE ; .1288888454437255859375000000000
C000B728 F04932BC 49EB6AA9 4CEB3F91  0520           .OCTA   ^XC000B728F04932BC49EB6AA94CEB3F91 ; .2504608670053217096790625535 61
25BE7132 2800B628 00F2F59E D6084003  0530           .OCTA   ^X25BE71322800B62800F2F59ED6084003 ; .7344296840954295810636208603 37
                                     0540   ; Entry  2
00000000 00000000 00007C00 3FEC3FFE  0540           .OCTA   ^X000000000000000000007C003FEC3FFE ; .1562127768993377685546875000000
2000D92B 6C94EDA2 C4E13F70 9A6CBF92  0550           .OCTA   ^X2000D92B6C94EDA2C4E13F709A6CBF92 ;-.6175356587835251976067325519 17
6BF2E1D1 2BD82203 F6BD4E03 1C1D4004  0560           .OCTA   ^X6BF2E1D12BD82203F6BD4E031C1D4004 ; .8878577239744036343747946594 70
                                     0570   ; Entry  3
00000000 00000000 0000D800 8FC93FFE  0570           .OCTA   ^X00000000000000000000D8008FC93FFE ; .1952092051506042480468750000000
E000B9B3 D2C2448D FFEC00AE 272F3F92  0580           .OCTA   ^XE000B9B3D2C2448DFFEC00AE272F3F92 ; .4441429237374547983580963356 23
728CB36C 241CBEC2 C9DCD558 61764004  0590           .OCTA   ^X728CB36C241CBEC2C9DCD55861764004 ; .1104575602857121183290750899 97
                                     05A0   ; Entry  4
00000000 00000000 0000A200 FF873FFE  05A0           .OCTA   ^X00000000000000000000A200FF873FFE ; .2497704178094863891601562500000
6000B408 356B3E7F 4E0A9D7D 6112BF92  05B0           .OCTA   ^X6000B408356B3E7F4E0A9D7D6112BF92 ;-.5312447777480676479186161255 13
8DFD9216 C4765F2A BDA97B3E C0C34004  05C0           .OCTA   ^X8DFD9216C4765F2ABDA97B3EC0C34004 ; .1402386247877192806033489369 73
                                     05D0   ; Entry  5
00000000 00000000 00002E00 3FB73FFF  05D0           .OCTA   ^X000000000000000000002E003FB73FFF ; .3122222125530242919921875000000
F000CBCD 5D4B1DF5 DDD84230 8B2ABF93  05E0           .OCTA   ^XF000CBCD5D4B1DF5DDD842308B2ABF93 ;-.1189156634515228440718123549 45
D1D87411 BE7A13C6 CC03B026 156E4005  05F0           .OCTA   ^XD1D87411BE7A13C6CC03B026156E4005 ; .1733952345995948446962663796 20
                                     0600   ; Entry  6
00000000 00000000 00001C00 8F3D3FFF  0600           .OCTA   ^X000000000000000000001C008F3D3FFF ; .3898815512657165527343750000000
2000FBCD 63B12926 7F10AEAD A8B5BF93  0610           .OCTA   ^X2000FBCD63B129267F10AEADA8B5BF93 ;-.1278064253739077715281429826 16
5977A2A2 C3076FEE 67EE5C53 54CC4005  0620           .OCTA   ^X5977A2A2C3076FEE67EE5C5354CC4005 ; .2129989273624860500359355230 19
                                     0630   ; Entry  7
00000000 00000000 00006C00 FE673FFF  0630           .OCTA   ^X000000000000000000006C00FE673FFF ; .4984413981437683105468750000000
40005A63 7B69C202 D1C6A8EF FF6F3F91  0640           .OCTA   ^X40005A637B69C202D1C6A8EFFF6F3F91 ; .3847618114745689823815794840 52
3A973C90 E63C687C 0826A50B A7E54005  0650           .OCTA   ^X3A973C90E63C687C0826A50BA7E54005 ; .2649356560048399888704653545 62
                                     0660   ; Entry  8
00000000 00000000 0000D600 4DF14000  0660           .OCTA   ^X00000000000000000000D6004DF14000 ; .6522356867790222167968750000000
9800A1AF 979C621A 79EF938D AD5D3F94  0670           .OCTA   ^X9800A1AF979C621A79EF938DAD5D3F94 ; .2584149801756739568584789423 30
E41B2F9C 4C7B9C23 52421D20 08E94006  0680           .OCTA   ^XE41B2F9C4C7B9C2352421D2008E94006 ; .3311382508517301743195460536 10
                                     0690   ; Entry  9
00000000 00000000 00002400 BC0E4000  0690           .OCTA   ^X000000000000000000002400BC0E4000 ; .8672953844070434570312500000000
0000BF3F 0245F876 17B6076E C1F53F91  06A0           .OCTA   ^X0000BF3F0245F87617B6076EC1F53F91 ; .3385101322349548486702179159 00
30D96B97 B7CA0E60 EF49C626 477A4006  06B0           .OCTA   ^X30D96B97B7CA0E60EF49C626477A4006 ; .4093494825761548056[illegible]463077332
                                     06C0   ; Entry 10
00000000 00000000 0000B000 2B974001  06C0           .OCTA   ^X00000000000000000000B0002B974001 ; .1170283317565917968750000000000
B00048B3 B1D17F3F 234C0829 91CC3F94  06D0           .OCTA   ^XB00048B3B1D17F3F234C082991CC3F94 ; .2418227727953071984458188936 91
7CFE3ABE 7F365421 A4E4F78B 8BE34006  06E0           .OCTA   ^X7CFE3ABE7F365421A4E4F78B8BE34006 ; .4948631199929199381541848825 13
                                     06F0   ; Entry 11
```

```
00000000 00000000 0000D600 A51B4001  06F0           .OCTA    ^X00000000000000000000D600A51B4001 ; .16449559926986694335937500000
C000F599 3C7B4FC5 4D224704 35393F92  0700           .OCTA    ^XC000F5993C7B4FC54D22470435393F92 ; .46526817199302666764435974513
1B3E2B18 61DDD13A 677E5B33 D5A14006  0710           .OCTA    ^X1B3E2B1861DDD13A677E5B33D5A14006 ; .58703787232967308375133701580
                                     0720    ; Entry 12
00000000 00000000 00000A00 49104002  0720           .OCTA    ^X00000000000000000000A0049104002 ; .25708019733428955078125000000
20002563 930862FA 5A827DB3 1DA2BF94  0730           .OCTA    ^X20002563930862FA5A827DB31DA2BF94 ;-.17191015021934972963177883888
D049E728 F2987BC8 4135A6E1 12FA4007  0740           .OCTA    ^XD049E728F2987BC84135A6E112FA4007 ; .68744777221303021073019348712
                                     0750    ; Entry 13
00000000 00000000 00001C00 56FA4003  0750           .OCTA    ^X000000000000000000001C0056FA4003 ; .53590154647827148437500000000
F8006797 565D6070 0481B17F D4F53F95  0760           .OCTA    ^XF8006797565D60700481B17FD4F53F95 ; .56448975328670213188681058457
DD3FE7AF 556969FA FECD68FE 3DB84007  0770           .OCTA    ^XDD3FE7AF556969FAFECD68FE3DB84007 ; .79430088028242019840044649383
                                     0780
                                     0780    ;
                                     0780    ; Tables to be used in POLYH for computing HATAND:  HATANDTAB1 is obtained
                                     0780    ; by multiplying the coefficients of HATANTAB1 by 180/pi.  ATANDTAB2 is the
                                     0780    ; same as ATANDTAB1 except that C0 is set to 180/pi - 64 instead of 180/pi.
                                     0780    ;
                                     0780
                                     0780    HATANDTAB1:
E3427579 7A5A5278 8C4CBB23 DAA84001  0780           .OCTA    ^XE34275797A5A52788C4CBB23DAA84001       ; C14 =  0.18541371309077459
94C77AFA 69E9B386 3937BC21 0F26C002  0790           .OCTA    ^X94C77AFA69E9B3863937BC210F26C002       ; C13 = -0.21183695947860862
7A9E56C2 5E79404F 86A78356 25584002  07A0           .OCTA    ^X7A9E56C25E79404F86A7835625584002       ; C12 =  0.22917637035837348
826666F2 71466582 7AC30564 3EDDC002  07B0           .OCTA    ^X826666F2714665827AC305643EDDC002       ; C11 = -0.24911200276015065
613E9D28 B4C320D0 79083E26 5D3B4002  07C0           .OCTA    ^X613E9D28B4C320D079083E265D3B4002       ; C10 =  0.27283704460006115
B69362BB 9E37F59D 00081C56 81FEC002  07D0           .OCTA    ^XB69362BB9E37F59D00081C5681FEC002       ; C09 = -0.30155673427507917
C9A28575 BC9686FB B0F34CD8 AF674002  07E0           .OCTA    ^XC9A28575BC9686FBB0F34CD8AF674002       ; C08 =  0.33703399713575914
17F287FF A01B2906 EACC8A4A E8ECC002  07F0           .OCTA    ^X17F287FFA01B2906EACC8A4AE8ECC002       ; C07 = -0.38197186342054874
2345AC31 BD3D8710 C2894FC8 1A124003  0800           .OCTA    ^X2345AC31BD3D8710C2894FC81A124003       ; C06 =  0.44073676548524862
35291286 F556CE32 2BB97590 4D5BC003  0810           .OCTA    ^X35291286F556CE322BB975904D5BC003       ; C05 = -0.52087072284620291
A3969252 F66134E8 C3AAC893 976F4003  0820           .OCTA    ^XA3969252F66134E8C3AAC893976F4003       ; C04 =  0.63661977236758134
483F83ED 79D12203 225B6EA8 05ECC004  0830           .OCTA    ^X483F83ED79D12203225B6EA805ECC004       ; C03 = -0.81851113590117601
65F98598 77582F9E 301967B8 6EB14004  0840           .OCTA    ^X65F9859877582F9E301967B86EB14004       ; C02 =  0.11459155902616464
ABBDC4A9 B8C9A7AE D2BFD66E 3193C005  0850           .OCTA    ^XABBDC4A9B8C9A7AED2BFD66E3193C005       ; C01 = -0.19098593171027440
81A5A6FE 152E7B86 3C1FC1A6 CA5D4006  0860           .OCTA    ^X81A5A6FE152E7B863C1FC1A6CA5D4006       ; C00 =  0.57295779513082320
                            0000000F  0870    HATANDLEN1 = .- HATANDTAB1/16
                                     0870
                                     0870    HATANDTAB2:
E3427579 7A5A5278 8C4CBB23 DAA84001  0870           .OCTA    ^XE34275797A5A52788C4CBB23DAA84001       ; C14 =  0.18541371309077459
94C77AFA 69E9B386 3937BC21 0F26C002  0880           .OCTA    ^X94C77AFA69E9B3863937BC210F26C002       ; C13 = -0.21183695947860862
7A9E56C2 5E79404F 86A78356 25584002  0890           .OCTA    ^X7A9E56C25E79404F86A7835625584002       ; C12 =  0.22917637035837348
826666F2 71466582 7AC30564 3EDDC002  08A0           .OCTA    ^X826666F2714665827AC305643EDDC002       ; C11 = -0.24911200276015065
613E9D28 B4C320D0 79083E26 5D3B4002  08B0           .OCTA    ^X613E9D28B4C320D079083E265D3B4002       ; C10 =  0.27283704460006115
B69362BB 9E37F59D 00081C56 81FEC002  08C0           .OCTA    ^XB69362BB9E37F59D00081C5681FEC002       ; C09 = -0.30155673427507917
C9A28575 BC9686FB B0F34CD8 AF674002  08D0           .OCTA    ^XC9A28575BC9686FBB0F34CD8AF674002       ; C08 =  0.33703399713575914
17F287FF A01B2906 EACC8A4A E8ECC002  08E0           .OCTA    ^X17F287FFA01B2906EACC8A4AE8ECC002       ; C07 = -0.38197186342054874
2345AC31 BD3D8710 C2894FC8 1A124003  08F0           .OCTA    ^X2345AC31BD3D8710C2894FC81A124003       ; C06 =  0.44073676548524862
35291286 F556CE32 2BB97590 4D5BC003  0900           .OCTA    ^X35291286F556CE322BB975904D5BC003       ; C05 = -0.52087072284620291
A3969252 F66134E8 C3AAC893 976F4003  0910           .OCTA    ^XA3969252F66134E8C3AAC893976F4003       ; C04 =  0.63661977236758134
483F83ED 79D12203 225B6EA8 05ECC004  0920           .OCTA    ^X483F83ED79D12203225B6EA805ECC004       ; C03 = -0.81851113590117601
65F98598 77582F9E 301967B8 6EB14004  0930           .OCTA    ^X65F9859877582F9E301967B86EB14004       ; C02 =  0.11459155902616464
ABBDC4A9 B8C9A7AE D2BFD66E 3193C005  0940           .OCTA    ^XABBDC4A9B8C9A7AED2BFD66E3193C005       ; C01 = -0.19098593171027440
                                     0950    H_PI_OV_180_M_64:
F2DBC80B 568A23CF 1F04F2CE AD11C003  0950           .OCTA    ^XF2DBC80B568A23CF1F04F2CEAD11C003       ; C00 = -0.67042204869176791
                            0000000F  0960    HATANDLEN2 = .- HATANDTAB2/16
                                     0960
                                     0960
                                     0960    H_90:
```

```
00000000 00000000 00000000 68004007  0960   368           .OCTA   ^X00000000000000000000000068004007      ; 90.
                                     0970   369 H_M90:
00000000 00000000 00000000 6800C007  0970   370           .OCTA   ^X0000000000000000000000006800C007      ; -90.
                                     0980   371 H_180:
00000000 00000000 00000000 68004008  0980   372           .OCTA   ^X00000000000000000000000068004008      ; 180
```

```
            0990          .SBTTL  MTH$HATAN - Standard H Floating Point Arc Tangent
            0990
            0990
            0990    ;++
            0990    ; FUNCTIONAL DESCRIPTION:
            0990    ;
            0990    ; HATAN - H floating point arctangent function
            0990    ;
            0990    ; HATAN is computed using the following steps:
            0990    ;
            0990    ;    1. If X > 11 then
            0990    ;       a. Let W = 1/X.
            0990    ;       b. Compute HATAN(W) = W*P(W**2), where P is a polynomial of
            0990    ;          degree 13.
            0990    ;       c. Set HATAN(X) = pi/2 - HATAN(W)
            0990    ;    2. If 3/32 =< X =< 11 then
            0990    ;       a. Obtain XHI by table look-up.
            0990    ;       b. Compute Z = (X - XHI)/(1 + X*XHI).
            0990    ;       c. Compute HATAN(Z) = Z*P(Z**2), where P is a polynomial of
            0990    ;          degree 13.
            0990    ;       d. Obtain HATAN(XHI) by table look-up.  HATAN(XHI) will have
            0990    ;          two parts - the high order bits, HATAN_XHI_HI, and the low
            0990    ;          order bits, HATAN_XHI_LO.
            0990    ;       e. Compute HATAN(X) = HATAN_XHI_HI + (HATAN_XHI_LO + HATAN(Z)).
            0990    ;    3. If 0 =< X < 3/32 then
            0990    ;       a. Compute HATAN(X) = X + X*Q(X**2), where Q is a polynomial
            0990    ;          of degree 13.
            0990    ;    4. If X < 0 then
            0990    ;       a. Compute Y = HATAN(|X|) using steps 1 to 3.
            0990    ;       b. Set HATAN(X) = -Y.
            0990    ;
            0990    ;
            0990    ; CALLING SEQUENCE:
            0990    ;
            0990    ;       Arctangent.wh.v = MTH$HATAN(x.rh.r)
            0990    ;
            0990    ;               -or-
            0990    ;
            0990    ;       CALL MTH$HATAN(Arctangent.wh.r, x.rh.r)
            0990    ;
            0990    ;       Because an H-floating result cannot be expressed in 64 bits, it is
            0990    ;       returned as the first argument, with the input parameter displaced
            0990    ;       to the second argument, in accordance with the Procedure Calling
            0990    ;       Standard.
            0990    ;
            0990    ; INPUT PARAMETERS:
            0990    ;
00000004    0990            LONG = 4                        ; Define longword multiplier
00000008    0990            x = 2 * LONG                    ; X is the argument
            0990    ;
            0990    ; IMPLICIT INPUTS:      none
            0990    ;
            0990    ; OUTPUT PARAMETERS:
            0990    ;
            0990    ;       VALUE:  H floating point arctangent (in radians) of the argument
            0990    ;
00000004    0990            Arctangent = 1 * LONG           ; Arctangent is the result
```

```
                              0990  ;
                              0990  ; IMPLICIT OUTPUTS:     none
                              0990  ;
                              0990  ; SIDE EFFECTS:
                              0990  ;
                              0990  ; Signals:      none
                              0990  ;
                              0990  ; NOTE: This procedure disables floating point underflow, enables
                              0990  ; integer overflow, causes no floating overflow or other arithmetic
                              0990  ; traps, and preserves enables across the call.
                              0990  ;
                              0990  ;---
                              0990
                              0990
                        41FC  0990        .ENTRY  MTH$HATAN, ACMASK       ; Standard call-by-reference entry
                              0992                                        ; Disable DV (and FU), enable IV
                              0992        MTH$FLAG_JACKET                 ; Flag that this is a jacket procedure in
                              0992
      6D   00000000'GF    9E  0992        MOVAB   G^MTH$$JACKET_HND, (FP)
                              0999                                        ; set handler address to jacket
                              0999                                        ; handler
                              0999
                              0999                                        ; case of an error in special JSB routine
         50    08 BC 70FD     0999        MOVH    @x(AP), R0              ; R0/R3 = arg
                 0089    30   099E        BSBW    MTH$HATAN_R8            ; Call special HATAN routine
         04 BC    50 7DFD     09A1        MOVO    R0, @Arctangent(AP)     ; Store result in first argument
                          04  09A6        RET                             ; Return to caller
                              09A7
```

```
                            09A7         .SBTTL  MTH$HATAN2 - Standard H Floating Point Arctangent With 2 Arguments
                            09A7  ;++
                            09A7  ; FUNCTIONAL DESCRIPTION:
                            09A7  ;
                            09A7  ; HATAN2  - H floating point arctangent function
                            09A7  ;
                            09A7  ; HATAN2(X,Y) is computed as following:
                            09A7  ;
                            09A7  ;       If Y = 0 or X/Y > 2**114, HATAN2(X,Y) = PI/2 * (sign X)
                            09A7  ;       If Y > 0 and X/Y =< 2**114, HATAN2(X,Y) = HATAN(X/Y)
                            09A7  ;       If Y < 0 and X/Y =< 2**114, HATAN2(X,Y) = PI * (sign X) + HATAN(X/Y)
                            09A7  ;
                            09A7  ;
                            09A7  ; CALLING SEQUENCE:
                            09A7  ;
                            09A7  ;       Arctangent2.wh.v = MTH$HATAN2(x.rh.r, y.rh.r)
                            09A7  ;
                            09A7  ;               -or-
                            09A7  ;
                            09A7  ;       CALL MTH$HATAN2(Arctangent2.wh.r, x.rh.r, y.rh.r)
                            09A7  ;
                            09A7  ;       Because an H floating result cannot be expressed in 64 bits, it is
                            09A7  ;       returned as the first argument, with the input parameter displaced
                            09A7  ;       to the second argument, in accordance with the Procedure Calling
                            09A7  ;       Standard.
                            09A7  ;
                            09A7  ; INPUT PARAMETERS:
                            09A7
                   00000008 09A7         x = 2 * LONG                    ; X is the first argument
                   0000000C 09A7         y = 3 * LONG                    ; Y is the second argument
                            09A7
                            09A7  ; SIDE EFFECTS: See description of MTH$HATAN
                            09A7  ;
                            09A7  ; OUTPUT PARAMETERS:
                            09A7  ;
                   00000004 09A7         Arctangent2 = 1 * LONG          ; Arctangent2 is the result
                            09A7  ;
                            09A7  ;--
                            09A7
                            09A7
                       41FC 09A7         .ENTRY  MTH$HATAN2, ACMASK      ; Standard call-by-reference entry
                            09A9                                         ; Disable DV (and FU), enable IV
                            09A9         MTH$FLAG_JACKET                 ; Flag that this is a jacket procedure in
                            09A9
6D     00000000'GF       9E 09A9         MOVAB   G^MTH$$JACKET_HND, (FP)
                            09B0                                         ; set handler address to jacket
                            09B0                                         ; handler
                            09B0
                            09B0                                         ; Case of an error in special JSB routine
       50    08 BC 70FD     09B0         MOVH    @x(AP), R0              ; R0/R3 = arg1
       54    0C BC 70FD     09B5         MOVH    @y(AP), R4              ; R4/R7 = arg2
                            09BA  ;
                            09BA  ; Test if Y = 0 or X/Y > 2**114
                            09BA  ;
                  42     13 09BA         BEQL    INF                     ; Branch to INF if Y = 0
58     50    8000 8F     AB 09BC         BICW3   #^X8000, R0, R8         ; R8 = exponent(X)
7E     54    8000 8F     AB 09C2         BICW3   #^X8000, R4, -(SP)      ; Stack = exponent(Y)
```

```
         58   8E   A2 09C8            SUBW2   (SP)+, R8              ; R8 = exponent(X) - exponent(Y)
   0073 8F    58   B1 09CB            CMPW    R8, #115               ; Compare R8 with 115
              2C   14 09D0            BGTR    INF                    ; If X/Y > 2**114, branch to INF
                      09D2    ;
                      09D2    ; Test if Y > 0 or Y < 0
                      09D2    ;
              54   B5 09D2            TSTW    R4                     ; Test the sign of Y
              20   14 09D4            BGTR    A2PLUS                 ; Branch to A2PLUS if Y > 0
              50   B5 09D6            TSTW    R0                     ; Test the sign of X
              0E   18 09D8            BGEQ    A1PLUS                 ; Branch to A1PLUS if X >= 0
                      09DA    ;
                      09DA    ; Y < 0 and X < 0 and X/Y =< 2**114
                      09DA    ;
              4A   10 09DA            BSBB    MTH$HATAN_R8D          ; R0/R3 = HATAN(X/Y)
   50    FAAF CF 62FD 09DC            SUBH2   H_PI, R0               ; R0/R3 = -PI + HATAN(X/Y)
     04 BC     50 7DFD 09E2           MOVO    R0, @Arctangent2(AP)   ; Store result in first argument
                   04 09E7            RET                            ; Return to caller
                      09E8    ;
                      09E8    ; Y < 0 and X > 0 and X/Y =< 2**114
                      09E8    ;
                      09E8    A1PLUS:
              3C   10 09E8            BSBB    MTH$HATAN_R8D          ; R0/R3 = HATAN(X/Y)
   50    FAA1 CF 60FD 09EA            ADDH2   H_PI, R0               ; R0/R3 = PI + HATAN(X/Y)
     04 BC     50 7DFD 09F0           MOVO    R0, @Arctangent2(AP)   ; Store result in first argument
                   04 09F5            RET                            ; Return
                      09F6    ;
                      09F6    ; Y > 0 and X/Y =< 2**114
                      09F6    ;
                      09F6    A2PLUS:
              2E   10 09F6            BSBB    MTH$HATAN_R8D          ; R0/R3 = HATAN(X/Y)
     04 BC     50 7DFD 09F8           MOVO    R0, @Arctangent2(AP)   ; Store result in first argument
                   04 09FD            RET                            ; Return
                      09FE    ;
                      09FE    ; Y = 0 or X/Y > 2**114
                      09FE    ;
                      09FE    INF:
              50   B5 09FE            TSTW    R0                     ; Test the sign of X
              0A   14 0A00            BGTR    1$                     ; Branch if X > 0
              10   13 0A02            BEQL    2$                     ; Branch if X = 0
  04 BC   FAA7 CF 7DFD 0A04           MOVO    H_MPI_OVER_2, @Arctangent2(AP)
                      0A0B                                           ; Store result in first argument
                   04 0A0B            RET                            ; Return
                      0A0C
  04 BC   FA8F CF 7DFD 0A0C   1$:     MOVO    H_PI_OVER_2, @Arctangent2(AP)
                      0A13                                           ; Store result in first argument
                   04 0A13            RET                            ; Return
                      0A14
                      0A14    ;+
                      0A14    ; Here if both X = 0 and Y = 0.  Signal INVALID ARG TO MATH LIBRARY
                      0A14    ;-
                      0A14
   50    01   0F   79 0A14    2$:     ASHQ    #15, #1, R0            ; R0/R3 = reserved operand, copied
                      0A18                                           ; to CHF$_MCH_SAVR0/R1 so handlers
                      0A18                                           ; can change if they want to continue.
              52   7C 0A18            CLRQ    R2                     ; ...
      7E   00'8F   9A 0A1A            MOVZBL  #MTH$K_INVARGMAT, -(SP) ; Code for invalid argument to math library
 00000000'GF  01   FB 0A1E            CALLS   #1, G^MTH$$SIGNAL      ; Signal SEVERE error
```

```
04  0A25   564          RET                                   ; Return if a handler says SS$_CONTINUE
```

MTH
Pse

PSE

.
_MT

Pha

Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
195
The
113
1 p

Mac

_$2

0 G

The

MAC

```
                          0A26         .SBTTL  MTH$HATAN_R8 - Special HATAN routine
                          0A26
                          0A26  ; Special HATAN - used by the standard routine, and directly.
                          0A26  ;
                          0A26  ; CALLING SEQUENCE:
                          0A26  ;       save anything needed in R0:R8
                          0A26  ;       MOVH    ..., R0                 ; Input in R0/R3
                          0A26  ;       JSB     MTH$HATAN_R8
                          0A26  ;       return with result in R0/R3
                          0A26
                          0A26  ; Note: This routine is written to avoid causing any integer overflows, floating
                          0A26  ; overflows, or floating underflows or divide by 0 conditions, whether enabled or
                          0A26  ; not.
                          0A26  ;
                          0A26  ; REGISTERS USED:
                          0A26  ;       R0/R3 - floating argument then result
                          0A26  ;       R4/R7 - scratch
                          0A26  ;       R8    - Pointer into HATAN_TABLE
                          0A26  ;       R0:R5 - POLYH
                          0A26  ;       stack - Y during POLYH
                          0A26
                          0A26
                          0A26
                          0A26 MTH$HATAN_R8D:                               ; For local use only!
          50    54 66FD   0A26         DIVH2   R4, R0
                          0A2A MTH$HATAN_R8::                               ; Special HATAN routine
               50 73FD    0A2A         TSTH    R0                           ; R0 = X = argument
               03    18   0A2D         BGEQ    POS_ARG                      ;
             0087    31   0A2F         BRW     NEG_ARG                      ; Branch to negative argument logic
                          0A32  ;
                          0A32  ; Argument is positive
                          0A32  ;
                          0A32 POS_ARG:
      58    50    03  9C  0A32         ROTL    #3, R0, R8                   ; R8 = Biased exponent bits and first
  58   FFF80000 8F    CA  0A36         BICL    #^XFFF80000, R8              ;    three fraction bits
  58   0001FFEC 8F    C2  0A3D         SUBL    #^X1FFEC, R8                 ; Argument is less than 3/32,
                  4C  19  0A44         BLSS    SMALL                        ;    branch to small argument logic
            58    37  D1  0A46         CMPL    #^X37, R8                    ; Argument is greater than of equal to
                  4A  15  0A49         BLEQ    LARGE_ARG                    ;    11, branch to large argument logic
                          0A4B  ;
                          0A4B  ; Logic for positive medium sized arguments.  Get pointer into HATAN_TABLE.
                          0A4B  ;
  57   00000000'GF    DE  0A4B         MOVAL   G^MTH$$AB_ATAN_V, R7         ; R7 = Address of RTL vector entry
  57   00000000'GF    C0  0A52         ADDL    G^MTH$$AB_ATAN_V, R7         ; R7 = Address of MTH$$AB_ATAN table
         58   6748    90  0A59         MOVB    (R7)[R8], R8                 ; R8 = Index into HATAN_TABLE.
    58   F5AD CF48 7EFD   0A5D         MOVAO   HATAN_TABLE[R8], R8          ; R8 = pointer to XHI
                          0A64  ;
                          0A64  ; Compute Z
                          0A64  ;
          54    88 7DFD   0A64         MOVO    (R8)+, R4                    ; R4 = XHI
     7E   50    54 63FD   0A68         SUBH3   R4, R0, -(SP)                ; (SP) = X - XHI
          50    54 64FD   0A6D         MULH2   R4, R0                       ; R0 = X*XHI
          50    08 60FD   0A71         ADDH2   #1, R0                       ; R0 = 1 + X*XHI
          6E    50 66FD   0A75         DIVH2   R0, (SP)                     ; (SP) = Z = (X - XHI)/(1 + X*XHI)
                          0A79  ;
                          0A79  ; Evaluate Z*P(Z**2)
                          0A79  ;
```

```
          50     6E     6E 65FD  0A79           MULH3   (SP), (SP), R0            ; R0 = Z**2
  F82B CF        0E     50 75FD  0A7E           POLYH   R0, #HATANLEN1-1, HATANTAB1
                                 0A85                                             ; R0 = P(Z**2)
                 50     8E 64FD  0A85           MULH2   (SP)+, R0                 ; R0 = HATAN(Z) = Z*P(Z**2)
                 50     88 60FD  0A89           ADDH2   (R8)+, R0                 ; R0 = HATAN_XHI_LO + HATAN(Z)
                 50     68 60FD  0A8D           ADDH2   (R8), R0                  ; R0 = HATAN(X) = HATAN_XHI_HI +
                                 0A91                                             ;     (HATAN_XHI_LO + HATAN(Z))
                            05   0A91           RSB                               ; Return
                                 0A92
                                 0A92
                    00A9    31   0A92   SMALL:  BRW     SMALL_ARG                 ; Dummy label used to avoid adding
                                 0A95                                             ;   an extra insrtuction in the
                                 0A95                                             ;   medium argument logic
                                 0A95   ;
                                 0A95   ; Large positive argument logic.
                                 0A95   ;
                                 0A95
                                 0A95   LARGE_ARG:
 7E    F565 CF          50 67FD  0A95           DIVH3   R0, H_M1.0, -(SP)         ; (SP) = -W = -1/X
          50     6E     6E 65FD  0A9C           MULH3   (SP), (SP), R0            ; R0 = W**2
  F808 CF        0E     50 75FD  0AA1           POLYH   R0, #HATANLEN1-1, HATANTAB1
                                 0AA8                                             ; R0 = P(W**2)
                 50     8E 64FD  0AA8           MULH2   (SP)+, R0                 ; R0 = HATAN(W) = -W*P(W**2)
          50     FA1F   CF 60FD  0AAC           ADDH2   H_PI_OVER_2_LO, R0        ;
          50     FA09   CF 60FD  0AB2           ADDH2   H_PI_OVER_2_HI, R0        ; R0 = HATAN(X) = PI/2 - HATAN(W)
                            05   0AB8           RSB                               ; Return
                                 0AB9
                                 0AB9   ;
                                 0AB9   ; Logic for negative arguments
                                 0AB9   ;
                                 0AB9
                                 0AB9   NEG_ARG:
          58     50     03    9C 0AB9           ROTL    #3, R0, R8                ; R8 = Biased exponent bits and first
     58    FFF80000 8F        CA 0ABD           BICL    #^XFFF80000, R8           ;    three fraction bits
     58    0005FFEC 8F        C2 0AC4           SUBL    #^X5FFEC, R8              ; Argument is less than 3/32,
                        C5    19 0ACB           BLSS    SMALL                     ;    branch to small argument logic
                 58     37    D1 0ACD           CMPL    #^X37, R8                 ; Argument is greater than of equal to
                        48    15 0AD0           BLEQ    N_LARGE_ARG               ;    11, branch to large argument logic
                                 0AD2   ;
                                 0AD2   ; Logic for negative medium sized arguments.  Get index into HATAN_TABLE.
                                 0AD2   ;
     57    00C00000'GF        DE 0AD2           MOVAL   G^MTH$$AB_ATAN_V, R7      ; R7 = Address of RTL vector entry
     57    00000000'GF        C0 0AD9           ADDL    G^MTH$$AB_ATAN_V, R7      ; R7 = Address of MTH$$AB_ATAN table
              58   6748       90 0AE0           MOVB    (R7)[R8], R8              ; R8 = Index into HATAN_TABLE.
      58    F526 CF48       7EFD 0AE4           MOVAO   HATAN_TABLE[R8], R8       ; R8 = pointer to XHI
                                 0AEB   ;
                                 0AEB   ; Compute Z
                                 0AEB   ;
                 54     88 7DFD  0AEB           MOVO    (R8)+, R4                 ; R4 = XHI
          7E     50     54 61FD  0AEF           ADDH3   R4, R0, -(SP)             ; (SP) = X + XHI = X - (-XHI)
                 50     54 64FD  0AF4           MULH2   R4, R0                    ; R0 = X*XHI
          50     08     50 63FD  0AF8           SUBH3   R0, #1, R0                ; R0 = 1 - X*XHI = 1 + X*(-XHI)
                 6E     50 66FD  0AFD           DIVH2   R0, (SP)                  ; (SP) = Z = (X-(-XHI))/(1+X*(-XHI))
                                 0B01   ;
                                 0B01   ; Evaluate Z*P(Z**2)
                                 0B01   ;
          50     6E     6E 65FD  0B01           MULH3   (SP), (SP), R0            ; R0 - Z**2
```

```
F7A3 CF    0E    50 75FD  0B06          POLYH   R0, #HATANLEN1-1, HATANTAB1
                          0B0D                                                  ; R0 = P(Z**2)
           50    8E 64FD  0B0D          MULH2   (SP)+, R0                       ; R0 = HATAN(Z) = Z*P(Z**2)
           50    88 62FD  0B11          SUBH2   (R8)+, R0                       ; R0 = HATAN_XHI_LO + HATAN(Z)
           50    68 62FD  0B15          SUBH2   (R8), R0                        ; R0 = HATAN(X) = HATAN_XHI_HI +
                          0B19                                                  ;       (HATAN_XHI_LO + HATAN(Z))
                      05  0B19          RSB                                     ; Return
                          0B1A  ;
                          0B1A  ; Logic for large negative  arguments
                          0B1A  ;
                          0B1A
                          0B1A  N_LARGE_ARG:
7E    F4E0 CF    50 67FD  0B1A          DIVH3   R0, H_M1.0, -(SP)               ; (SP) = W = 1/|X|
      50   6E    6E 65FD  0B21          MULH3   (SP), (SP), R0                  ; R0 = W**2
F783 CF    0E    50 75FD  0B26          POLYH   R0, #HATANLEN1-1, HATANTAB1
                          0B2D                                                  ; R0 = P(W**2)
           50    8E 64FD  0B2D          MULH2   (SP)+, R0                       ; R0 = HATAN(W) = W*P(W**2)
      50   F99A CF 62FD   0B31          SUBH2   H_PI_OVER_2_LO, R0              ;
      50   F984 CF 52FD   0B37          SUBH2   H_PI_OVER_2_HI, R0              ; R0 = HATAN(X) = HATAN(W) - PI/2
                      05  0B3D          RSB                                     ; Return
                          0B3E
                          0B3E  ;
                          0B3E  ; Small argument logic.
                          0B3E  ;
                          0B3E
                          0B3E  SMALL_ARG:
           7E    50 7DFD  0B3E          MOVO    R0, -(SP)                       ; (SP) = argument = X
      50   8000 8F    AA  0B42          BICW    #^X8000, R0                     ; R0 = |X|
      50   3FC9 8F    B1  0B47          CMPW    #^X3FC9, R0                     ; Compare 2^-56 to |X|
           05         19  0B4C          BLSS    1$                              ; Needs polynomial evaluation
           50    8E 7DFD  0B4E          MOVO    (SP)+, R0                       ; Return with answer equal
                      05  0B52          RSB                                     ;    to argument
                          0B53
           50    50 64FD  0B53  1$:     MULH2   R0, R0                          ; R0 = X**2
F842 CF    0E    50 75FD  0B57          POLYH   R0, #HATANLEN2-1, HATANTAB2
                          0B5E                                                  ; R0 = Q(X**2)
           50    6E 64FD  0B5E          MULH2   (SP), R0                        ; R0 = X*Q(X**2)
           50    8E 60FD  0B62          ADDH2   (SP)+, R0                       ; R0 = HATAN(X) = X + X*Q(X**2)
                      05  0B66          RSB                                     ; Return
                          0B67
```

```
          0B67         .SBTTL  MTH$HATAND - Standard H Floating Point Arc Tangent
          0B67
          0B67
          0B67   ;++
          0B67   ; FUNCTIONAL DESCRIPTION:
          0B67   ;
          0B67   ; HATAND - H floating point arctangent function
          0B67   ;
          0B67   ; HATAND is computed using the following steps:
          0B67   ;
          0B67   ;    1. If X > 11 then
          0B67   ;         a. Let W = 1/X.
          0B67   ;         b. Compute HATAND(W) = W*P(W**2), where P is a polynomial of
          0B67   ;            degree 13.
          0B67   ;         c. Set HATAND(X) = pi/2 - HATAND(W)
          0B67   ;    2. If 3/32 =< X =< 11 then
          0B67   ;         a. Obtain XHI by table look-up.
          0B67   ;         b. Compute Z = (X - XHI)/(1 + X*XHI).
          0B67   ;         c. Compute HATAND(Z) = Z*P(Z**2), where P is a polynomial of
          0B67   ;            degree 13.
          0B67   ;         d. Obtain HATAND(XHI) by table look-up.  HATAND(XHI) will have
          0B67   ;            two parts - the high order bits, HATAND_XHI_HI, and the low
          0B67   ;            order bits, HATAND_XHI_LO.
          0B67   ;         e. Compute HATAND(X) = HATAND_XHI_HI + (HATAND_XHI_LO + HATAND(Z)).
          0B67   ;    3. If 0 =< X < 3/32 then
          0B67   ;         a. Compute HATAND(X) = X + X*Q(X**2), where Q is a polynomial
          0B67   ;            of degree 13.
          0B67   ;    4. If X < 0 then
          0B67   ;         a. Compute Y = HATAND(|X|) using steps 1 to 3.
          0B67   ;         b. Set HATAND(X) = -Y.
          0B67   ;
          0B67   ;
          0B67   ; CALLING SEQUENCE:
          0B67   ;
          0B67   ;       Arctangent.wh.v = MTH$HATAND(x.rh.r)
          0B67   ;
          0B67   ;               -or-
          0B67   ;
          0B67   ;       CALL MTH$HATAND(Arctangent.wh.r, x.rh.r)
          0B67   ;
          0B67   ;       Because an H-floating result cannot be expressed in 64 bits, it is
          0B67   ;       returned as the first argument, with the input parameter displaced
          0B67   ;       to the second argument, in accordance with the Procedure Calling
          0B67   ;       Standard.
          0B67   ;
          0B67   ; INPUT PARAMETERS:
          0B67   ;
00000004  0B67         LONG = 4                        ; Define longword multiplier
00000008  0B67         x = 2 * LONG                    ; X is the argument
          0B67   ;
          0B67   ; IMPLICIT INPUTS:      none
          0B67   ;
          0B67   ; OUTPUT PARAMETERS:
          0B67   ;
          0B67   ;       VALUE:  H floating point arctangent (in degrees) of the argument
          0B67   ;
00000004  0B67         Arctangent = 1 * LONG           ; Arctangent is the result
```

```
                          0B67   778 ;
                          0B67   779 ; IMPLICIT OUTPUTS:     none
                          0B67   780 ;
                          0B67   781 ; SIDE EFFECTS:
                          0B67   782 ;
                          0B67   783 ; Signals:      none
                          0B67   784 ;
                          0B67   785 ; NOTE: This procedure disables floating point underflow, enables
                          0B67   786 ; integer overflow, causes no floating overflow or other arithmetic
                          0B67   787 ; traps, and preserves enables across the call.
                          0B67   788 ;
                          0B67   789 ;---
                          0B67   790
                          0B67   791
                    41FC  0B67   792         .ENTRY  MTH$HATAND, ACMASK      ; STANDdard call-by-reference entry
                          0B69   793                                         ; Disable DV (and FU), enable IV
                          0B69   794         MTH$FLAG_JACKET                 ; Flag that this is a jacket procedure in
                          0B69
6D    00000000'GF   9E    0B69               MOVAB   G^MTH$$JACKET_HND, (FP)
                          0B70                                               ; set handler address to jacket
                          0B70                                               ; handler
                          0B70
                          0B70   795                                         ; case of an error in special JSB routine
      50    08 BC 70FD    0B70   796         MOVH    @x(AP), R0              ; R0/R3 = arg
            0089    30    0B75   797         BSBW    MTH$HATAND_R8           ; Call special HATAND routine
   04 BC    50 7DFD       0B78   798         MOVO    R0, @Arctangent(AP)     ; Store result in first argument
                    04    0B7D   799         RET                             ; Return to caller
                          0B7E   800
```

```
                          0B7E          .SBTTL  MTH$HATAND2 - Standard H Floating Point Arctangent With 2 Arguments
                          0B7E   ;++
                          0B7E   ; FUNCTIONAL DESCRIPTION:
                          0B7E   ;
                          0B7E   ; HATAND2  - H floating point arctangent function
                          0B7E   ;
                          0B7E   ; HATAND2(X,Y) is computed as following:
                          0B7E   ;
                          0B7E   ;      If Y = 0 or X/Y > 2**114, HATAND2(X,Y) = 90 * (sign X)
                          0B7E   ;      If Y > 0 and X/Y =< 2**114, HATAND2(X,Y) = HATAND(X/Y)
                          0B7E   ;      If Y < 0 and X/Y =< 2**114, HATAND2(X,Y) = 180 * (sign X) + HATAND(X/Y)
                          0B7E   ;
                          0B7E   ;
                          0B7E   ; CALLING SEQUENCE:
                          0B7E   ;
                          0B7E   ;      Arctangent2.wh.v = MTH$HATAND2(x.rh.r, y.rh.r)
                          0B7E   ;
                          0B7E   ;              -or-
                          0B7E   ;
                          0B7E   ;      CALL MTH$HATAND2(Arctangent2.wh.r, x.rh.r, y.rh.r)
                          0B7E   ;
                          0B7E   ;      Because an H floating result cannot be expressed in 64 bits, it is
                          0B7E   ;      returned as the first argument, with the input parameter displaced
                          0B7E   ;      to the second argument, in accordance with the Procedure Calling
                          0B7E   ;      Standard.
                          0B7E   ;
                          0B7E   ; INPUT PARAMETERS:
                          0B7E
                 00000008 0B7E          x = 2 * LONG                    ; X is the first argument
                 0000000C 0B7E          y = 3 * LONG                    ; Y is the second argument
                          0B7E
                          0B7E   ; SIDE EFFECTS: See description of MTH$HATAND
                          0B7E   ;
                          0B7E   ; OUTPUT PARAMETERS:
                          0B7E   ;
                 00000004 0B7E          Arctangent2 = 1 * LONG          ; Arctangent2 is the result
                          0B7E   ;
                          0B7E   ;--
                          0B7E
                          0B7E
                     41FC 0B7E          .ENTRY  MTH$HATAND2, ACMASK     ; STANDdard call-by-reference entry
                          0B80                                          ; Disable DV (and FU), enable IV
                          0B80          MTH$FLAG_JACKET                 ; Flag that this is a jacket procedure in
                          0B80
  6D    00000000'GF    9E 0B80          MOVAB   G^MTH$$JACKET_HND, (FP)
                          0B87                                          ; set handler address to jacket
                          0B87                                          ; handler
                          0B87
                          0B87                                          ; Case of an error in special JSB routine
     50    08 BC 70FD     0B87          MOVH    @x(AP), R0              ; R0/R3 = arg1
     54    0C BC 70FD     0B8C          MOVH    @y(AP), R4              ; R4/R7 = arg2
                          0B91   ;
                          0B91   ; Test if Y = 0 or X/Y > 2**114
                          0B91   ;
                 42    13 0B91          BEQL    INF_DEG                 ; Branch to INF_DEG if Y = 0
 58   50    8000 8F    AB 0B93          BICW3   #^X8000, R0, R8         ; R8 = exponent(X)
 7E   54    8000 8F    AB 0B99          BICW3   #^X8000, R4, -(SP)      ; Stack = exponent(Y)
```

```
          58    8E    A2  0B9F        SUBW2   (SP)+, R8               ; R8 = exponent(X) - exponent(Y)
     0073 8F    58    B1  0BA2        CMPW    R8, #115                ; Compare R8 with 115
                2C    14  0BA7        BGTR    INF_DEG                 ; If X/Y > 2**114, branch to INF_DEG
                          0BA9   ;
                          0BA9   ; Test if Y > 0 or Y < 0
                          0BA9   ;
                54    B5  0BA9        TSTW    R4                      ; Test the sign of Y
                20    14  0BAB        BGTR    A2PLUSD                 ; Branch to A2PLUSD if Y > 0
                50    B5  0BAD        TSTW    R0                      ; Test the sign of X
                0E    18  0BAF        BGEQ    A1PLUSD                 ; Branch to A1PLUSD if X >= 0
                          0BB1   ;
                          0BB1   ; Y < 0 and X < 0 and X/Y =< 2**114
                          0BB1   ;
                4A    10  0BB1        BSBB    MTH$HATAND_R8D          ; R0/R3 = HATAND(X/Y)
     50   FDC8 CF   62FD  0BB3        SUBH2   H_180, R0               ; R0/R3 = -180 + HATAND(X/Y)
       04 BC    50  7DFD  0BB9        MOVO    R0, @Arctangent2(AP)    ; Store result in first argument
                      04  0BBE        RET                             ; Return to caller
                          0BBF   ;
                          0BBF   ; Y < 0 and X > 0 and X/Y =< 2**114
                          0BBF   ;
                          0BBF   A1PLUSD:
                3C    10  0BBF        BSBB    MTH$HATAND_R8D          ; R0/R3 = HATAND(X/Y)
     50   FDBA CF   60FD  0BC1        ADDH2   H_180, R0               ; R0/R3 = 180 + HATAND(X/Y)
       04 BC    50  7DFD  0BC7        MOVO    R0, @Arctangent2(AP)    ; Store result in first argument
                      04  0BCC        RET                             ; Return
                          0BCD   ;
                          0BCD   ; Y > 0 and X/Y =< 2**114
                          0BCD   ;
                          0BCD   A2PLUSD:
                2E    10  0BCD        BSBB    MTH$HATAND_R8D          ; R0/R3 = HATAND(X/Y)
       04 BC    50  7DFD  0BCF        MOVO    R0, @Arctangent2(AP)    ; Store result in first argument
                      04  0BD4        RET                             ; Return
                          0BD5   ;
                          0BD5   ; Y = 0 or X/Y > 2**114
                          0BD5   ;
                          0BD5   INF_DEG:
                50    B5  0BD5        TSTW    R0                      ; Test the sign of X
                0A    14  0BD7        BGTR    1$                      ; Branch if X > 0
                10    13  0BD9        BEQL    2$                      ; Branch if X = 0
  04 BC      FD90 CF  7DFD  0BDB      MOVO    H_M90, @Arctangent2(AP)
                          0BE2                                        ; Store result in first argument
                      04  0BE2        RET                             ; Return
                          0BE3
  04 BC      FD78 CF  7DFD  0BE3 1$:  MOVO    H_90, @Arctangent2(AP)
                          0BEA                                        ; Store result in first argument
                      04  0BEA        RET                             ; Return
                          0BEB
                          0BEB   ;+
                          0BEB   ; Here if both X = 0 and Y = 0.  Signal INVALID ARG TO MATH LIBRARY
                          0BEB   ;-
                          0BEB
     50    01    0F   79  0BEB   2$:  ASHQ    #15, #1, R0             ; R0/R3 = reserved operand, co180ed
                          0BEF                                        ; to CHF$_MCH_SAVR0/R1 so handlers
                          0BEF                                        ; can change if they want to continue.
                52    7C  0BEF        CLRQ    R2                      ; ...
       7E   00'8F    9A   0BF1        MOVZBL  #MTH$K_INVARGMAT, -(SP) ; Code for invalid argument to math library
  00000000'GF    01   FB  0BF5        CALLS   #1, G^MTH$$SIGNAL       ; Signal SEVERE error
```

```
04  0BFC    911          RET                        ; Return if a handler says SS$_CONTINUE
```

```
                          0BFD          .SBTTL  MTH$HATAND_R8 - Special HATAND routine
                          0BFD
                          0BFD   ; Special HATAND - used by the sTANDdard routine, and directly.
                          0BFD   ;
                          0BFD   ; CALLING SEQUENCE:
                          0BFD   ;       save anything needed in R0:R8
                          0BFD   ;       MOVH    ..., R0                 ; Input in R0/R3
                          0BFD   ;       JSB     MTH$HATAND_R8
                          0BFD   ;       return with result in R0/R3
                          0BFD
                          0BFD   ; Note: This routine is written to avoid causing any integer overflows, floating
                          0BFD   ; overflows, or floating underflows or divide by 0 conditions, whether enabled or
                          0BFD   ; not.
                          0BFD   ;
                          0BFD   ; REGISTERS USED:
                          0BFD   ;       R0/R3 - floating argument then result
                          0BFD   ;       R4/R7 - scratch
                          0BFD   ;       R8    - Pointer into HATAND_TABLE
                          0BFD   ;       R0:R5 - POLYH
                          0BFD   ;       stack - Y during POLYH
                          0BFD
                          0BFD
                          0BFD
                          0BFD  MTH$HATAND_R8D:                        ; For local use only!
           50    54 66FD  0BFD          DIVH2   R4, R0
                          0C01  MTH$HATAND_R8::                        ; Special HATAND routine
              50 73FD     0C01          TSTH    R0                     ; (SP) = X = argument
              03    18    0C04          BGEQ    POS_ARGD               ;
            0081    31    0C06          BRW     NEG_ARGD               ; Branch to negative argument logic
                          0C09   ;
                          0C09   ; Argument is positive
                          0C09   ;
                          0C09  POS_ARGD:
        58    50    03 9C 0C09          ROTL    #3, R0, R8             ; R8 = Biased exponent bits and first
  58   FFF80000 8F     CA 0C0D          BICL    #^XFFF80000, R8        ;    three fraction bits
  58   0001FFEC 8F     C2 0C14          SUBL    #^X1FFEC, R8           ; Argument is less than 3/32,
                  4C   19 0C1B          BLSS    SMALLD                 ;    branch to small argument logic
           58     37   D1 0C1D          CMPL    #^X37, R8              ; Argument is greater than of equal to
                  4A   15 0C20          BLEQ    LARGE_ARGD             ;    11, branch to large argument logic
                          0C22   ;
                          0C22   ; Logic for positive medium sized arguments.  Get pointer into HATAND_TABLE.
                          0C22   ;
  57   00000000'GF     DE 0C22          MOVAL   G^MTH$$AB_ATAN_V, R7   ; R7 = Address of RTL vector entry
  57   00000000'GF     C0 0C29          ADDL    G^MTH$$AB_ATAN_V, R7   ; R7 = Address of MTH$$AB_ATAN table
       58    6748      90 0C30          MOVB    (R7)[R8], R8           ; R8 = Index into HATAND_TABLE.
   58    F8A6 CF48   7EFD 0C34          MOVAO   HATAND_TABLE[R8], R8   ; R8 = pointer to XHI
                          0C3B   ;
                          0C3B   ; Compute z
                          0C3B   ;
           54     88 7DFD 0C3B          MOVO    (R8)+, R4              ; R4 = XHI
      7E   50     54 63FD 0C3F          SUBH3   R4, R0, -(SP)          ; (SP) = X - XHI
           50     54 64FD 0C44          MULH2   R4, R0                 ; R0 = X*XHI
           50     08 60FD 0C48          ADDH2   #1, R0                 ; R0 = 1 + X*XHI
           6E     50 66FD 0C4C          DIVH2   R0, (SP)               ; (SP) = Z = (X - XHI)/(1 + X*XHI)
                          0C50   ;
                          0C50   ; Evaluate Z*P(Z**2)
                          0C50   ;
```

```
          50    6E    6E 65FD  0C50            MULH3   (SP), (SP), R0          ; R0 = Z**2
     FB24 CF    0E    50 75FD  0C55            POLYH   R0, #HATANDLEN1-1, HATANDTAB1
                               0C5C                                            ; R0 = P(Z**2)
                50    8E 64FD  0C5C            MULH2   (SP)+, R0               ; R0 = HATAN(Z) = Z*P(Z**2)
                50    88 60FD  0C60            ADDH2   (R8)+, R0               ; R0 = HATAN_XHI_LO + HATAN(Z)
                50    68 60FD  0C64            ADDH2   (R8), R0                ; R0 = HATAN(X) = HATAN_XHI_HI +
                               0C68                                            ;      (HATAN_XHI_LO + HATAN(Z))
                          05   0C68            RSB                             ; Return
                               0C69
                               0C69
                    009D  31   0C69    SMALLD: BRW     SMALL_ARGD              ; Dummy label used to avoid adding
                               0C6C                                            ;   an extra insrtuction in the
                               0C6C                                            ;   medium argument logic
                               0C6C    ;
                               0C6C    ; Large positive argument logic.
                               0C6C    ;
                               0C6C
                               0C6C    LARGE_ARGD:
     7E   F38E CF     50 67FD  0C6C            DIVH3   R0, H_M1.0, -(SP)       ; (SP) = -W = -1/X
          50    6E    6E 65FD  0C73            MULH3   (SP), (SP), R0          ; R0 = W**2
     FB01 CF    0E    50 75FD  0C78            POLYH   R0, #HATANDLEN1-1, HATANDTAB1
                               0C7F                                            ; R0 = P(W**2)
                50    8E 64FD  0C7F            MULH2   (SP)+, R0               ; R0 = -HATAND(Z) = -Z*P(W**2)
          50    FCD8 CF 60FD   0C83            ADDH2   H_90, R0                ; R0 = HATAND(X) = 90 - HATAND(Z)
                          05   0C89            RSB                             ; Return
                               0C8A
                               0C8A    ;
                               0C8A    ; Logic for negative arguments
                               0C8A    ;
                               0C8A
                               0C8A    NEG_ARGD:
          58    50    03  9C   0C8A            ROTL    #3, R0, R8              ; R8 = Biased exponent bits and first
      58   FFF80000 8F    CA   0C8E            BICL    #^XFFF80000, R8         ;     three fraction bits
      58   0005FFEC 8F    C2   0C95            SUBL    #^X5FFEC, R8            ; Argument is less than 3/32,
                     6B   19   0C9C            BLSS    SMALL_ARGD              ;     branch to small argument logic
                58   37   D1   0C9E            CMPL    #^X37, R8               ; Argument is greater than of equal to
                     48   15   0CA1            BLEQ    N_LARGE_ARGD            ;      11, branch to large argument logic
                               0CA3    ;
                               0CA3    ; Logic for negative medium sized arguments.  Get index into HATAND_TABLE.
                               0CA3    ;
      57   00000000'GF    DE   0CA3            MOVAL   G^MTH$$AB_ATAN_V, R7    ; R7 = Address of RTL vector entry
      57   00000000'GF    C0   0CAA            ADDL    G^MTH$$AB_ATAN_V, R7    ; R7 = Address of MTH$$AB_ATAN table
            58   6748     90   0CB1            MOVB    (R7)[R8], R8            ; R8 = Index into HATAND_TABLE.
       58   F825 CF48 7EFD     0CB5            MOVAO   HATAND_TABLE[R8], R8    ; R8 = pointer to XHI
                               0CBC    ;
                               0CBC    ; Compute Z
                               0CBC    ;
                54    88 7DFD  0CBC            MOVO    (R8)+, R4               ; R4 = XHI
          7E    50    54 61FD  0CC0            ADDH3   R4, R0, -(SP)           ; R0 = X + XHI = X - (-XHI)
                50    54 64FD  0CC5            MULH2   R4, R0                  ; R0 = X*XHI
          50    08    50 63FD  0CC9            SUBH3   R0, #1, R0              ; R4 = 1 - X*XHI = 1 + X*(-XHI)
                6E    50 66FD  0CCE            DIVH2   R0, (SP)                ; (SP) = Z
                               0CD2    ;
                               0CD2    ; Evaluate Z*P(Z**2)
                               0CD2    ;
          50    6E    6E 65FD  0CD2            MULH3   (SP), (SP), R0          ; R0 = Z**2
     FAA2 CF    0E    50 75FD  0CD7            POLYH   R0, #HATANDLEN1-1, HATANDTAB1
```

```
                          0CDE                                          ; R0 = P(Z**2)
            50   8E 64FD  0CDE          MULH2   (SP)+, R0               ; R0 = HATAND(Z) = Z*P(Z**2)
            50   88 62FD  0CE2          SUBH2   (R8)+, R0               ; R0 = HATAND_XHI_LO + HATAND(Z)
            50   68 62FD  0CE6          SUBH2   (R8), R0                ; R0 = HATAND(X) = HATAND_XHI_HI +
                          0CEA                                          ;       (HATAND_XHI_LO + HATAND(Z))
                      05  0CEA          RSB                             ; Return
                          0CEB  ;
                          0CEB  ; Logic for large negative  arguments
                          0CEB  ;
                          0CEB
                          0CEB  N_LARGE_ARGD:
7E   F30F CF   50 67FD    0CEB          DIVH3   R0, H_M1.0, -(SP)       ; (SP) = W = 1/|X|
     50   6E   6E 65FD    0CF2          MULH3   (SP), (SP), R0          ; R0 = W**2
FA82 CF   0E   50 75FD    0CF7          POLYH   R0, #HATANDLEN1-1, HATANDTAB1
                          0CFE                                          ; R0 = P(W**2)
            50   8E 64FD  0CFE          MULH2   (SP)+, R0               ; R0 = HATAND(W) = W*P(W**2)
     50   FC59 CF 62FD    0D02          SUBH2   H_90, R0                ; R0 = HATAND(X) = HATAND(W) - 90
                      05  0D08          RSB                             ; Return
                          0D09
                          0D09  ;
                          0D09  ; Small argument logic.
                          0D09  ;
                          0D09
                          0D09  SMALL_ARGD:
            7E   50 70FD  0D09          MOVH    R0, -(SP)               ; (SP) = argument = X
                 2C   13  0D0D          BEQL    3$                      ;
     50   8000 8F     AA  0D0F          BICW    #^X8000, R0             ; R0 = |X|
     50   3FC9 8F     B1  0D14          CMPW    #^X3FC9, R0             ; Compare 2^-56 to |X|
                 09   19  0D19          BLSS    1$                      ; Needs polynomial evaluation
50   6E   FC30 CF 65FD    0D1B          MULH3   H_PI_OV_180_M_64, (SP), R0
                          0D22                                          ; R0 = X*(pi/180 - 64)
                 0F   11  0D22          BRB     2$                      ; Argument = 0 = return value
            50   50 64FD  0D24  1$:     MULH2   R0, R0                  ; R0 = X**2
FB41 CF   0E   50 75FD    0D28          POLYH   R0, #HATANDLEN2-1, HATANDTAB2
                          0D2F                                          ; R0 = Q(X**2)
            50   6E 64FD  0D2F          MULH2   (SP), R0                ; R0 = X*Q(X**2)
            6E   06   A0  0D33  2$:     ADDW    #^X6, (SP)              ; (SP) = X*2**6
            50   8E 60FD  0D36          ADDH2   (SP)+, R0               ; R0 = HATAND(X) = X*2**6 + X*Q(X**2)
                      05  0D3A          RSB                             ; Return
                          0D3B
                 8E 73FD  0D3B  3$:     TSTH    (SP)+                   ; Remove argument from stack
                      05  0D3E          RSB                             ; Return
                          0D3F
                          0D3F          .END
```

```
A1PLUS                  000009E8 R     01
A1PLUSD                 00000BBF R     01
A2PLUS                  000009F6 R     01
A2PLUSD                 00000BCD R     01
ACMASK                = 000041FC
ARCTANGENT            = 00000004
ARCTANGENT2           = 00000004
HATANDLEN1            = 0000000F
HATANDLEN2            = 0000000F
HATANDTAB1              00000780 R     01
HATANDTAB2              00000870 R     01
HATAND_TABLE            000004E0 R     01
HATANLEN1             = 0000000F
HATANLEN2             = 0000000F
HATANTAB1               000002B0 R     01
HATANTAB2               000003A0 R     01
HATAN_TABLE             00000010 R     01
H_180                   00000980 R     01
H_90                    00000960 R     01
H_M1.0                  00000000 R     01
H_M90                   00000970 R     01
H_MPI_OVER_2            000004B0 R     01
H_PI                    00000490 R     01
H_PI_OVER_2             000004A0 R     01
H_PI_OVER_2_HI          000004C0 R     01
H_PI_OVER_2_LO          000004D0 R     01
H_PI_OV_180_M_64        00000950 R     01
INF                     000009FE R     01
INF_DEG                 00000BD5 R     01
LARGE_ARG               00000A95 R     01
LARGE_ARGD              00000C6C R     01
LONG                  = 00000004
MTH$$AB_ATAN_V          ********    X  00
MTH$$JACKET_HND         ********    X  01
MTH$$SIGNAL             ********    X  00
MTH$HATAN               00000990 RG    01
MTH$HATAN2              000009A7 RG    01
MTH$HATAND              00000B67 RG    01
MTH$HATAND2             00000B7E RG    01
MTH$HATAND_R8           00000C01 RG    01
MTH$HATAND_R8D          00000BFD R     01
MTH$HATAN_R8            00000A2A RG    01
MTH$HATAN_R8D           00000A26 R     01
MTH$K_INVARGMAT         ********    X  00
NEG_ARG                 00000AB9 R     01
NEG_ARGD                00000C8A R     01
N_LARGE_ARG             00000B1A R     01
N_LARGE_ARGD            00000CEB R     01
POS_ARG                 00000A32 R     01
POS_ARGD                00000C09 R     01
SMALL                   00000A92 R     01
SMALLD                  00000C69 R     01
SMALL_ARG               00000B3E R     01
SMALL_ARGD              00000D09 R     01
X                     = 00000008
Y                     = 0000000C
```

```
                                        +----------------+
                                        ! 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                 00000D3F  (  3391.)  01 (   1.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD  NOWRT NOVEC LONG

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

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                    29    00:00:00.06    00:00:00.86
Command processing               115    00:00:00.69    00:00:04.63
Pass 1                           134    00:00:02.89    00:00:09.91
Symbol table sort                  0    00:00:00.01    00:00:00.04
Pass 2                           215    00:00:02.36    00:00:07.11
Symbol table output                7    00:00:00.06    00:00:00.18
Psect synopsis output              3    00:00:00.02    00:00:00.06
Cross-reference output             0    00:00:00.00    00:00:00.00
Assembler run totals             505    00:00:06.10    00:00:22.79

The working set limit was 1200 pages.
19511 bytes (39 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 8 local symbols.
1130 source lines were read in Pass 1, producing 27 object records in Pass 2.
1 page of virtual memory was used to define 1 macro.

                                    +--------------------------+
                                    ! 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$:MTHHATAN/OBJ=OBJ$:MTHHATAN MSRC$:MTHJACKET/UPDATE=(ENH$:MTHJACKET)+MSRC
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

MTHGSIGN
LIS

MTHHCOSH
LIS

MTHHACOS
LIS

MTHGPROD
LIS

MTHGSINCO
LIS

MTHHASIN
LIS

MTHHATAN
LIS

MTHGTAN
LIS

MTHGTANH
LIS

MTHGSQRT
LIS

MTHHEXP
LIS

MTHGSINH
LIS

MTHHATANH
LIS