```
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMM          MMM  TTTTTTTTTTTTTTT   HHH          HHH  RRRRRRRRRRRR    TTTTTTTTTTTTTTT   LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMMMMM    MMMMMM        TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM   MMM   MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRR           TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHHHHHHHHHHHHHHH  RRRRRRRRRRRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR   RRR             TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR      RRR          TTT         LLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
MMM         MMM         TTT         HHH          HHH  RRR         RRR       TTT         LLLLLLLLLLLLLLLL
```

FILEID**OTSPOWHLU

```
  000000    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      000000    WW      WW  HH      HH  LL          UU      UU
  000000    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      000000    WW      WW  HH      HH  LL          UU      UU
00      00      TT      SS          PP      PP  00      00  WW      WW  HH      HH  LL          UU      UU
00      00      TT      SS          PP      PP  00      00  WW      WW  HH      HH  LL          UU      UU
00      00      TT      SS          PP      PP  00      00  WW      WW  HH      HH  LL          UU      UU
00      00      TT      SS          PP      PP  00      00  WW      WW  HH      HH  LL          UU      UU
00      00      TT        SSSSSS    PPPPPPPP    00      00  WW      WW  HHHHHHHHHH  LL          UU      UU
00      00      TT        SSSSSS    PPPPPPPP    00      00  WW      WW  HHHHHHHHHH  LL          UU      UU
00      00      TT              SS  PP          00      00  WW  WW  WW  HH      HH  LL          UU      UU
00      00      TT              SS  PP          00      00  WW  WW  WW  HH      HH  LL          UU      UU
00      00      TT              SS  PP          00      00  WWWW  WWWW  HH      HH  LL          UU      UU      ....
00      00      TT              SS  PP          00      00  WWWW  WWWW  HH      HH  LL          UU      UU      ....
  000000        TT      SSSSSSSS    PP            000000    WW      WW  HH      HH  LLLLLLLLLL  UUUUUUUUUU      ....
  000000        TT      SSSSSSSS    PP            000000    WW      WW  HH      HH  LLLLLLLLLL  UUUUUUUUUU      ....

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     1         .TITLE  OTS$POWHLU - H_floating ** unsigned integer power routine
0000     2         .IDENT  /1-001/                 ; File: OTSPOWHLU.MAR Edit: SBL1001
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: Language support library, Mathematics division
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       This module contains OTS$POWHLU_R3, a procedure which raises an
0000    35 ;       H_floating value to an unsigned integer power.
0000    36 ;
0000    37 ; ENVIRONMENT: Runs at any access mode, AST Reentrant
0000    38 ;
0000    39 ; AUTHOR: Steven B. Lionel, CREATION DATE: 22-JUL-1981
0000    40 ;
0000    41 ; MODIFIED BY:
0000    42 ;
0000    43 ; 1-001 - Original. SBL 22-JUL-1981
0000    44 ;--
```

```
                        .SBTTL  DECLARATIONS
                ;
                ; LIBRARY MACRO CALLS:
                ;
                        $SSDEF                  ; System error codes
                        $CHFDEF                 ; Define condition handler symbols.
                        $SFDEF                  ; Define stack frame symbols.
                        $PSLDEF                 ; Define program status longword
                                                ; symbols.
                ;
                ; EXTERNAL DECLARATIONS:
                ;
                        .DSABL  GBL             ; Force all external symbols to be declared
                        .EXTRN  MTH$K_UNDEXP    ; Undefined exponentiation
                        .EXTRN  MTH$K_FLOOVEMAT ; Floating overflow in Math Library
                        .EXTRN  MTH$K_FLOUNDMAT ; Floating underflow in Math Library
                        .EXTRN  MTH$$SIGNAL     ; Math error routine
                ;
                ; MACROS:
                ;
                        NONE
                ;
                ; EQUATED SYMBOLS:
                ;
                        NONE
                ;
                ; OWN STORAGE:
                ;
                        NONE
                ;
                ; PSECT DECLARATIONS:
                ;
00000000                .PSECT _OTS$CODE PIC, USR, CON, REL, LCL, SHR, -
                                        EXE, RD, NOWRT, LONG
```

```
                     0000     82          .SBTTL  OTS$POWHLU_R3 - H_floating ** unsigned integer power routine
                     0000     83 ;++
                     0000     84 ; FUNCTIONAL DESCRIPTION:
                     0000     85 ;
                     0000     86 ;       Raise an H_floating value to an unsigned integer power giving
                     0000     87 ;       a H_floating result in registers R0-R3.
                     0000     88 ;
                     0000     89 ;       The result is given by:
                     0000     90 ;
                     0000     91 ;       base    exponent        result
                     0000     92 ;
                     0000     93 ;       any     > 0             product (base * 2**i) where i is each
                     0000     94 ;                               non-zero bit position in exponent
                     0000     95 ;
                     0000     96 ;       > 0     = 0             1.0
                     0000     97 ;       = 0     = 0             Undefined exponentation
                     0000     98 ;       < 0     = 0             1.0
                     0000     99 ;
                     0000    100 ;
                     0000    101 ; CALLING SEQUENCE:
                     0000    102 ;
                     0000    103 ;       result.wh.v = OTS$POWHLU (base.rh.v, exponent.rlu.v)
                     0000    104 ;
                     0000    105 ; FORMAL PARAMETERS:
                     0000    106 ;
                     0000    107
           00000004  0000    108          base = 4        ; H_floating base.  Note that this is passed by
                     0000    109                          ; immediate value in four argument list positions,
                     0000    110                          ; which is a violation of the calling standard.
                     0000    111                          ; This is allowed for language support procedures.
                     0000    112
           00000014  0000    113          exponent = 20   ; Unsigned longword integer exponent
                     0000    114
                     0000    115 ;
                     0000    116 ; IMPLICIT INPUTS:
                     0000    117 ;
                     0000    118 ;       NONE
                     0000    119 ;
                     0000    120 ; IMPLICIT OUTPUTS:
                     0000    121 ;
                     0000    122 ;       NONE
                     0000    123 ;
                     0000    124 ; ROUTINE VALUE:
                     0000    125 ;
                     0000    126 ;       The base raised to the exponent's power.  This is returned in
                     0000    127 ;       registers R0-R3, which is a violation of the calling standard,
                     0000    128 ;       but is permitted for language support procedures.
                     0000    129 ;
                     0000    130 ; SIDE EFFECTS:
                     0000    131 ;
                     0000    132 ;       May signal:
                     0000    133 ;               MTH$_FLOOVEMAT - floating overflow in Math Library
                     0000    134 ;               MTH$_FLOUNDMAT - floating underflow in Math Library, only
                     0000    135 ;                                if the caller has FU enabled.
                     0000    136 ;               MTH$_UNDEXP    - undefined exponentiation, if both base and
                     0000    137 ;                                exponent are zero.
                     0000    138 ;
```

```
                                0000   139 ;--
                                0000   140
                          01FC  0000   141         .ENTRY OTS$POWHLU_R3, ^M<R2,R3,R4,R5,R6,R7,R8>  ; Entry point
                                0002   142
                6D   6F'AF    9E  0002   143         MOVAB   B^EXC_HANDLER, (FP)     ; Translate exceptions to MTH$ errors
                   0040 8F    B8  0006   144         BISPSW  #PSL$M_FU               ; Enable floating underflow detection
                   50   08 70FD  000A   145         MOVH    #1, R0                  ; R0-R3 = initial result
                54   04 AC 70FD  000E   146         MOVH    base(AP), R4            ; R4-R7 = base
                58   14 AC   D0  0013   147         MOVL    exponent(AP), R8        ; R8 = exponent
                        06   12  0017   148         BNEQ    BEGIN                   ; Skip if not zero
                        52 73FD  0019   149         TSTH    R2                      ; Is base also zero?
                        23   13  001C   150         BEQL    UNDEFINED               ; If so, error.  Otherwise, answer is
                                001E   151                                         ; 1.0.
                             04 001E   152         RET                             ; With result of 1.0
                                001F   153
                                001F   154 ;+
                                001F   155 ; Do the first iteration here so that we can clear the high-order exponent
                                001F   156 ; bit.  Afterwards, that bit will be zero so we don't have to worry about
                                001F   157 ; it.
                                001F   158 ;-
                                001F   159
                                001F   160 BEGIN:
                04 58   00   E5  001F   161         BBCC    #0, R8, 10$             ; Skip initial multiply
                   50   54 70FD  0023   162         MOVH    R4, R0                  ; New partial result
             58 58   FF 8F   9C  0027   163 10$:     ROTL    #-1, R8, R8             ; Get new low exponent bit and clear
                                002C   164                                         ; high bit
                        12   13  002C   165         BEQL    DONE                    ; Is that all?  If so, we're done.
                                002E   166
                                002E   167 ;+
                                002E   168 ; Each time we get here we know that there's at least one more exponent bit
                                002E   169 ; left, so square our current power of the base.
                                002E   170 ;-
                                002E   171
                                002E   172 LOOP:
                   54   54 64FD  002E   173         MULH2   R4, R4                  ; Square base.  This may overflow or
                                0032   174                                         ; underflow.  In either case, the
                                0032   175                                         ; final result would also overflow
                                0032   176                                         ; or underflow.  Our exception
                                0032   177                                         ; handler will catch these cases.
                   04 58   E9  0032   178         BLBC    R8, NEXT_BIT            ; See if this exponent bit is on.
                                0035   179                                         ; If so, multiply base by product.
                   50   54 64FD  0035   180         MULH2   R4, R0                  ; This too may overflow or underflow,
                                0039   181                                         ; which will be caught by the handler.
                                0039   182 NEXT_BIT:
             58 58   FF 8F   78  0039   183         ASHL    #-1, R8, R8             ; Get next bit of exponent.
                        EE   12  003E   184         BNEQ    LOOP                    ; Keep going if not all done.
                                0040   185
                                0040   186 DONE:
                             04 0040   187         RET                             ; Return with answer in R0-R3.
                                0041   188
                                0041   189 ;+
                                0041   190 ; Undefined exponentation error - 0**0
                                0041   191 ;-
                                0041   192
                                0041   193 UNDEFINED:
                7E   00'8F   9A  0041   194         MOVZBL  #MTH$K_UNDEXP, -(SP)    ; Indicate undefined exponentiation.
             5C   01   0F   79  0045   195         ASHQ    #15, #T, R0             ; R0-R3 = reserved floating
```

```
                52   7C  0049   196          CLRQ    R2                          ; operand
                18   11  004B   197          BRB     SIGNAL_ERROR
                         004D   198
                         004D   199 ;+
                         004D   200 ; EXC_HANDLER unwinds here if we got a floating overflow.
                         004D   201 ;-
                         004D   202
                         004D   203 OVERFLOW:
          7E    00'8F    9A  004D   204          MOVZBL  #MTH$K_FLOOVEMAT, -(SP) ; Floating overflow in Math Library
     50   01       0F    79  0051   205          ASHQ    #15, #1, R0             ; R0-R3 = reserved floating
                52   7C  0055   206          CLRQ    R2                          ; operand
                0C   11  0057   207          BRB     SIGNAL_ERROR
                         0059   208
                         0059   209 ;+
                         0059   210 ; EXC_HANDLER unwinds here if we got a floating underflow.
                         0059   211 ; If our caller enabled FU, then signal FLOUNDMAT, with a result of zero.
                         0059   212 ; Otherwise quietly return zero.
                         0059   213 ;-
                         0059   214
                         0059   215 UNDERFLOW:
                50 7CFD  0059   216          CLRH    R0                          ; Initial result
DF 04 AD        06   E1  005C   217          BBC     #PSL$V_FU, SF$W_SAVE_PSW(FP), DONE      ; Just return if
                         0061   218                                              ; caller disabled underflow.
          7E    00'8F    9A  0061   219          MOVZBL  #MTH$K_FLOUNDMAT, -(SP) ; Floating underflow in Math Library
                         0065   220 ;        BRB     SIGNAL_ERROR
                         0065   221
                         0065   222 ;+
                         0065   223 ; Signal a MTH$ error whose code has already been pushed on the stack.
                         0065   224 ; R0-R3 has either a reserved operand or a zero, depending on the
                         0065   225 ; exception type.
                         0065   226 ;-
                         0065   227
                         0065   228 SIGNAL_ERROR:
           6D    D4  0065   229          CLRL    (FP)                        ; Cancel our condition handler
00000000'GF     01   FB  0067   230          CALLS   #1, G^MTH$$SIGNAL           ; convert to 32-bit condition code
                         006E   231                                              ; and signal a MTH$ error.  Omit
                         006E   232                                              ; second argument to show that this
                         006E   233                                              ; is a CALL entry point.
                04   006E   234          RET                                 ; Return to caller with result in
                         006F   235                                              ; R0-R3
                         006F   236
```

```
                                006F   238 ;+
                                006F   239 ; EXC_HANDLER - This condition handler gets control if we got an exception
                                006F   240 ; while performing the exponentiation.  If it is a floating overflow,
                                006F   241 ; then continue execution at OVERFLOW, to cause MTH$_FLOOVEMAT to be
                                006F   242 ; signalled.  Similarly, continue at UNDERFLOW if an underflow was seen.
                                006F   243 ; Otherwise resignal.
                                006F   244 ;-
                                006F   245
                                006F   246 EXC_HANDLER:
                          0000  006F   247         .WORD   ^M<>                            ; Entry point
                                0071   248
          50    08 AC     D0    0071   249         MOVL    CHF$L_MCHARGLST(AP), R0 ; Get mechanism arguments list
                08 A0     D5    0075   250         TSTL    CHF$L_MCH_DEPTH(R0)             ; At depth zero?
                   16     12    0078   251         BNEQ    RESIGNAL                        ; If not, resignal
          50    04 AC     D0    007A   252         MOVL    CHF$L_SIGARGLST(AP), R0 ; Get signal arguments list
          51    04 A0     D0    007E   253         MOVL    CHF$L_SIG_NAME(R0), R1  ; Get signal name
     04B4 8F      51     B1    0082   254         CMPW    R1, #SS$_FLTOVF_F               ; Overflow fault?
                   0D     13    0087   255         BEQL    GOTO_OVERFLOW                   ; If so, continue at OVERFLOW
     04C4 8F      51     B1    0089   256         CMPW    R1, #SS$_FLTUND_F               ; Underflow fault?
                   11     13    008E   257         BEQL    GOTO_UNDERFLOW                  ; If so, continue at UNDERFLOW
                                0090   258 RESIGNAL:
          50    0918 8F     3C    0090   259         MOVZWL  #SS$_RESIGNAL, R0               ; Resignal exception
                   04     04    0095   260         RET                                     ; Return to VMS exception dispatcher
                                0096   261
                                0096   262 GOTO_OVERFLOW:
          51    60    D0      0096   263         MOVL    CHF$L_SIG_ARGS(R0), R1  ; Get number of signal arguments
  FC A041   B1 AF     DE      0099   264         MOVAL   B^OVERFLOW, -4(R0)[R1]  ; Move address of OVERFLOW routine
                                009F   265                                                 ; to PC in signal argument list
                   09     11    009F   266         BRB     CONTINUE                        ; Continue execution
                                00A1   267
                                00A1   268 GOTO_UNDERFLOW:
          51    60    D0      00A1   269         MOVL    CHF$L_SIG_ARGS(R0), R1  ; Get number of signal arguments
  FC A041   B2 AF     DE      00A4   270         MOVAL   B^UNDERFLOW, -4(R0)[R1] ; Move address of UNDERFLOW routine
                                00AA   271                                                 ; to PC in signal argument list
                                00AA   272 ;       BRB     CONTINUE                        ; Continue execution
                                00AA   273
                                00AA   274 CONTINUE:
          50    01    D0      00AA   275         MOVL    #SS$_CONTINUE, R0               ; Continue at PC in signal list
                   04     04    00AD   276         RET                                     ; Return to VMS exception dispatcher
                                00AE   277
                                00AE   278
                                00AE   279         .END                                    ; End of module OTS$POWHLU
```

```
BASE                            = 00000004
BEGIN                             0000001F R      02
CHF$L_MCHARGLST                 = 00000008
CHF$L_MCH_DEPTH                 = 00000008
CHF$L_SIGARGLST                 = 00000004
CHF$L_SIG_ARGS                  = 00000000
CHF$L_SIG_NAME                  = 00000004
CONTINUE                          000000AA R      02
DONE                              00000040 R      02
EXC_HANDLER                       0000006F R      02
EXPONENT                        = 00000014
GOTO_OVERFLOW                     00000096 R      02
GOTO_UNDERFLOW                    000000A1 R      02
LOOP                              0000002E R      02
MTH$$SIGNAL                       ******** *    X 00
MTH$K_FLOOVEMAT                   ******** *    X 00
MTH$K_FLOUNDMAT                   ******** *    X 00
MTH$K_UNDEXP                      ******** *    X 00
NEXT_BIT                          00000039 R      02
OTS$POWHLU_R3                     00000000 RG     02
OVERFLOW                          0000004D R      02
PSL$M_FU                        = 00000040
PSL$V_FU                        = 00000006
RESIGNAL                          00000090 R      02
SF$W_SAVE_PSW                   = 00000004
SIGNAL_ERROR                      00000065 R      02
SS$_CONTINUE                    = 00000001
SS$_FLTOVF_F                    = 00000484
SS$_FLTUND_F                    = 000004C4
SS$_RESIGNAL                    = 00000918
UNDEFINED                         00000041 R      02
UNDERFLOW                         00000059 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_OTS$CODE	000000AE (174.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.12	00:00:01.29
Command processing	115	00:00:00.59	00:00:05.10
Pass 1	206	00:00:05.01	00:00:13.88
Symbol table sort	0	00:00:00.77	00:00:01.22
Pass 2	61	00:00:01.13	00:00:02.99
Symbol table output	6	00:00:00.05	00:00:00.11
Psect synopsis output	2	00:00:00.03	00:00:00.05
Cross-reference output	0	00:00:00.00	00:00:00.00

Assembler run totals 427 00:00:07.70 00:00:24.70

The working set limit was 1200 pages.
26519 bytes (52 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 508 non-local and 1 local symbols.
279 source lines were read in Pass 1, producing 13 object records in Pass 2.
11 pages of virtual memory were used to define 10 macros.

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

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	7

562 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:OTSPOWHLU/OBJ=OBJ$:OTSPOWHLU MSRC$:OTSPOWHLU/UPDATE=(ENH$:OTSPOWHLU)

OTSPOWHH
LIS

OTSPOWII
LIS

OTSPOWRJ
LIS

UUXPOWGG
LIS

OTSPOWGLU
LIS

OTSPOWHLU
LIS

OTSPOWHJ
LIS

UUXPOWCJ
LIS

UUXPOWRR
LIS

OTSPOWLUL
LIS

OTSPOWRR
LIS

OTSPOWJJ
LIS

UUXEXP
LIS

UUXGSINCO
LIS

OTSPOWRLU
LIS