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
```

FILEID**OTSPOWGJ F 16

```
  OOOOOO    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      OOOOOO    WW      WW    GGGGGGGG          JJ
  OOOOOO    TTTTTTTTTT    SSSSSSSS  PPPPPPPP      OOOOOO    WW      WW    GGGGGGGG          JJ
OO      OO      TT      SS          PP      PP  OO      OO  WW      WW  GG                    JJ
OO      OO      TT      SS          PP      PP  OO      OO  WW      WW  GG                    JJ
OO      OO      TT      SS          PP      PP  OO      OO  WW      WW  GG                    JJ
OO      OO      TT      SS          PP      PP  OO      OO  WW      WW  GG                    JJ
OO      OO      TT        SSSSSS    PPPPPPPP    OO      OO  WW      WW  GG                    JJ
OO      OO      TT        SSSSSS    PPPPPPPP    OO      OO  WW      WW  GG                    JJ
OO      OO      TT              SS  PP          OO      OO  WW  WW  WW  GG  GGGGGG  JJ      JJ
OO      OO      TT              SS  PP          OO      OO  WW  WW  WW  GG  GGGGGG  JJ      JJ
OO      OO      TT              SS  PP          OO      OO  WWWW  WWWW  GG      GG  JJ      JJ      ....
OO      OO      TT              SS  PP          OO      OO  WWWW  WWWW  GG      GG  JJ      JJ      ....
  OOOOOO        TT      SSSSSSSS    PP            OOOOOO    WW      WW    GGGGGG      JJJJJJ        ....
  OOOOOO        TT      SSSSSSSS    PP            OOOOOO    WW      WW    GGGGGG      JJJJJJ        ....

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

Table of contents

```
0000             .TITLE  OTS$POWGJ - G REAL*8 ** INTEGER*4 power routine
0000             .IDENT  /1-005/         ; File: OTSPOWGJ.MAR  Edit: SBL1005
0000
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: Language support library - user callable
0000     ;++
0000     ; ABSTRACT:
0000     ;
0000     ;       G REAL*8 base to INTEGER*4 power.
0000     ;       Floating overflow and underflow can occur.
0000     ;       Undefined exponentation can occur if base is 0 and power
0000     ;       is 0 or negative.
0000     ;
0000     ;
0000     ;--
0000     ;
0000     ; VERSION: 1
0000     ;
0000     ; HISTORY:
0000     ; AUTHOR:
0000     ;       Steven B. Lionel, 6-Feb-79: Version 1
0000     ;
0000     ;
0000     ;
0000     ;
```

```
0000          .SBTTL  HISTORY         ; Detailed Current Edit History
0000
0000
0000    ; Edit History for Version 1 of OTS$POWGJ
0000    ; 1-002 SBL1002 - Declare externals.  SBL 17-May-1979
0000    ; 1-003 - Correct some comments.  JBS 30-JUL-1979
0000    ; 1-001 - Adapted from OTS$POWDJ version 1-001.  SBL 06-Feb-79
0000    ; 1-004 - Add handlers to catch SS$_FLTOVF and SS$_FLTDIV, and signal
0000    ;         MTH$_FLOOVEMAT or MTH$_FLOUNDMAT instead, depending on the context.
0000    ;         Also disable IV and change BLBS/ASHL at EXPGTR to BBSC/ROTL for
0000    ;         uniformity with OTS$POWRJ.  JAW 26-Feb-1980.
0000    ; 1-005 - Use general mode addressing.  SBL 30-Nov-1981
0000    ;
```

```
                    0000            .SBTTL  DECLARATIONS
                    0000
                    0000      ;
                    0000      ; INCLUDE FILES:
                    0000      ;
                    0000
                    0000      ;
                    0000      ; EXTERNAL SYMBOLS:
                    0000      ;
                    0000
                    0000            .DSABL  GBL
                    0000            .EXTRN  MTH$K_UNDEXP, MTH$K_FLOOVEMAT, MTH$K_FLOUNDMAT
                    0000            .EXTRN  MTH$$SIGNAL             ; Math error routine
                    0000            .EXTRN  SS$_FLTOVF, SS$_FLTOVF_F, SS$_FLTDIV, SS$_FLTDIV_F, SS$_CONTINUE
                    0000      ;
                    0000      ; MACROS:
                    0000      ;
                    0000            $CHFDEF                         ; Define condition handler symbols.
                    0000            $SFDEF                          ; Define stack frame symbols.
                    0000            $PSLDEF                         ; Define program status longword
                    0000                                            ; symbols.
                    0000
                    0000      ;
                    0000      ; EQUATED SYMBOLS:
                    0000      ;
        00000004    0000            base = 4                        ; base input formal - by-value
        0000000C    0000            exp = 12                        ; exponent intpu formal - by-value
                    0000                                            ; Note: G_floating by-value violates
                    0000                                            ; calling standard, but ok since this
                    0000                                            ; routine is a code support routine (OTS$)
                    0000
                    0000      ;
                    0000      ; OWN STORAGE:
                    0000      ;
                    0000
                    0000      ;
                    0000      ; PSECT DECLARATIONS:
                    0000      ;
                    0000
                00000000            .PSECT  _OTS$CODE PIC,SHR,LONG,EXE,NOWRT
                    0000                                            ; program section for OTS$ code
                    0000
```

```
0000            .SBTTL  OTS$POWGJ - G REAL*8 ** INTEGER*4
0000
0000      ;++
0000      ; FUNCTIONAL DESCRIPTION:
0000      ;
0000      ;       G_floating result = G_floating base ** signed longword exponent
0000      ;       The G_floating result is given by:
0000      ;
0000      ;       base    exponent        result
0000      ;
0000      ;       any     > 0             product (base * 2**i) where i is each
0000      ;                               non-zero bit position in exponent
0000      ;
0000      ;       > 0     = 0             1.0
0000      ;       = 0     = 0             Undefined exponentation
0000      ;       < 0     = 0             1.0
0000      ;
0000      ;       > 0     < 0             1.0 / product (base * 2**i)
0000      ;                               where i is each non-zero bit position
0000      ;                               in |exponent|
0000      ;       = 0     < 0             Undefined exponentation
0000      ;       < 0     < 0             1.0 / product (base * 2**i)
0000      ;                               where i is each non-zero bit position
0000      ;                               in |exponent|
0000      ;
0000      ;       Floating overflow can occur on either of the two MULG's.  If this
0000      ;       happens when the exponent is less than zero, the exception is caught by
0000      ;       a local condition handler named EXC_HNDLR_UNDER, which sets the result
0000      ;       to 0.0 and either signals MTH$_FLOUNDMAT (if FU is enabled in the
0000      ;       caller's PSW) or continues at POWGJX.  If it happens when the exponent
0000      ;       is greater than zero, the exception is caught by a local condition
0000      ;       handler named EXC_HNDLR_OVER, which sets the result to the reserved
0000      ;       operand (-0.0) and signals MTH$_FLOOVEMAT.
0000      ;
0000      ;       Floating overflow and floating divide by zero can occur on the DIVG.
0000      ;       These exceptions are caught by EXC_HNDLR_OVER, which sets the result to
0000      ;       the reserved operand (-0.0) and signals MTH$_FLOOVEMAT.
0000      ;
0000      ;       Undefined exponentiation occurs if base is 0 and
0000      ;       exponent is 0 or negative.
0000      ;
0000      ; CALLING SEQUENCE:
0000      ;
0000      ;       Power.wg.v = OTS$POWGJ (base.rg.v, exponent.rl.v)
0000      ;
0000      ; INPUT PARAMETERS:
0000      ;       base            - G REAL*8 base
0000      ;       exponent        - INTEGER*4 exponent
0000      ;
0000      ; IMPLICIT INPUTS:
0000      ;       The setting of FU in the caller's PSW.
0000      ;
0000      ; OUTPUT PARAMETERS:
0000      ;       NONE
0000      ;
0000      ; IMPLICIT OUTPUTS:
0000      ;       NONE
```

```
                      0000  ;
                      0000  ; FUNCTION VALUE:
                      0000  ;
                      0000  ;       G_floating base **  signed longword exponent
                      0000  ;
                      0000  ; SIDE EFFECTS:
                      0000  ;
                      0000  ;       Signals MTH$_FLOOVEMAT if floating overflow occurs on either of the two
                      0000  ;           MULG's when exponent > 0, or if floating overflow or divide by zero
                      0000  ;           occurs on the DIVG.
                      0000  ;       Signals MTH$_FLOUNDMAT if floating overflow occurs on either of the two
                      0000  ;           MULG's when exponent < 0 and caller has FU enabled.
                      0000  ;       SIGNALs MTH$_UNDEXP (82 = ' UNDEFINED EXPONENTATION') if
                      0000  ;       base is 0 and exponent is 0 or negative.
                      0000  ;
                      0000  ;--
                      0000
                      0000
                      0000
                001C  0000          .ENTRY  OTS$POWGJ, ^M<R2, R3, R4>
                      0002                                          ; Disable integer overflow.  (Occurs on
                      0002                                          ; maximum negative exponent.)
   6D   6C'AF     9E  0002          MOVAB   B^EXC_HNDLR_OVER, (FP)  ; Translate exceptions to
                      0006                                          ; MTH$_FLOOVEMAT.
     50    08   50FD  0006          MOVG    #1, R0                  ; R0/R1 = initial result
   52    04 AC  50FD  000A          MOVG    base(AP), R2            ; R2/R3 = base
   54    0C AC    D0  000F          MOVL    exp(AP), R4             ; R4 = exponent
            0E    14  0013          BGTR    EXPGTR                  ; branch if exponent > 0
   6D   5D'AF     9E  0015          MOVAB   B^EXC_HNDLR_UNDER, (FP) ; Translate exceptions to
                      0019                                          ; MTH$_FLOUNDMAT.
                      0019
            52  53FD  0019          TSTG    R2                      ; test base
            2F    13  001C          BEQL    UNDEFINED               ; undefined 0**0 or 0**(-n)
     54     54    CE  001E          MNEGL   R4, R4                  ; R4 = |exponent|
            29    13  0021          BEQL    POWGJX                  ; if exponent is 0, return R0 = 1.0
                      0023
                      0023  ;+
                      0023  ; Exponent is > 0 or (exponent is =< 0 and base is not = 0 -- use |exponent|)
                      0023  ;-
                      0023
   0C 54    00    E4  0023  EXPGTR: BBSC    #0, R4, PARTIAL         ; branch if |exponent| is odd
54 54    FF 8F    9C  0027  SQUAR:  ROTL    #-1, R4, R4             ; R4 = |exponent|/2
     52     52  44FD  002C  SQUAR1: MULG2   R2, R2                  ; R2/R3 = current power of base
                      0030                                          ; Floating overflow will trap or fault
                      0030                                          ; and signal SS$_FLTOVF or SS$_FLTOVF_F.
         F4 54    E9  0030          BLBC    R4, SQUAR               ; branch if next bit in |exponent| is 0
                      0033
                      0033  ;+
                      0033  ; Here when bit i of |exponent| is a 1.
                      0033  ; Partial result = partial result *  (base * 2**i)
                      0033  ;-
                      0033
                      0033  PARTIAL:
     50     52  44FD  0033          MULG2   R2, R0                  ; R0/R1 = new partial result
54 54    FF 8F    78  0037          ASHL    #-1, R4, R4             ; R4 = |exponent|/2
            EE    12  003C          BNEQ    SQUAR1                  ; loopback if more exponent bits are 1
                      003E
```

```
              0C AC    D5  003E            TSTL    exp(AP)                 ; test sign of exponent
                 09    14  0041            BGTR    POWGJX                  ; if exponent > 0, return R0
        6D    6C'AF    9E  0043            MOVAB   B^EXC_HNDLR_OVER, (FP)  ; Translate exceptions to
                           0047                                            ; MTH$_FLOOVEMAT.
      50    08    50 47FD  0047            DIVG3   R0, #1, R0              ; R0/R1 = 1.0/result
                       04  004C    POWGJX: RET                             ; return, result in R0
                           004D
                           004D    ;+
                           004D    ; Undefined exponentation error - 0**0 or 0**(-n)
                           004D    ;-
                           004D
                           004D    UNDEFINED:
      50    01    0F   79  004D            ASHQ    #15, #1, R0             ; R0/R1 = reserved floating operand
         7E    00'8F   9A  0051            MOVZBL  #MTH$K_UNDEXP, -(SP)    ; Indicate undefined exponentiation.
    00000000'GF   01   FB  0055            CALLS   #1, G^MTH$$SIGNAL       ; convert to 32-bit condition code
                           005C                                            ; and SIGNAL MTH$_UNDEXP
                           005C                                            ; Note: 2nd arg not needed since no JSB OTS$
                           005C                                            ; is possible.
                       04  005C            RET                             ; return
                           005D
                           005D    ;+
                           005D    ; The following handler is established to process exceptions which imply
                           005D    ; underflow of the final result (floating overflow in either of the two MULG's
                           005D    ; when exp < 0).  On the occurrence of such an exception, the handler signals
                           005D    ; MTH$_FLOUNDMAT.
                           005D    ;-
                           005D
                           005D    EXC_HNDLR_UNDER:
                 001C      005D            .WORD   ^M<R2, R3, R4>          ; Entry mask
                   2B  10  005F            BSBB    SETUP                   ; Set up R0:R3 and identify condition.
                           0061                                            ; Return only if FLTOVF or FLTDIV.
     1E 04 A2     06   E1  0061            BBC     #PSL$V_FU, SF$W_SAVE_PSW(R2), CON_U
                           0066                                            ; Branch if caller has not enabled FU.
        54     00'8F   9A  0066            MOVZBL  #MTH$K_FLOUNDMAT, R4    ; Report MTH$_FLOUNDMAT, not SS$_FLTOVF.
                   0C  11  006A            BRB     DO_SIG
                           006C
                           006C    ;+
                           006C    ; The following handler is established to process exceptions which imply
                           006C    ; overflow of the final result (floating overflow in either of the two MULG's
                           006C    ; when exp > 0, floating overflow in the DIVG, or floating divide by zero in the
                           006C    ; DIVG).  Or the occurrence of such an exception, the handler signals
                           006C    ; MTH$_FLOOVEMAT.
                           006C    ;-
                           006C
                           006C    EXC_HNDLR_OVER:
                 001C      006C            .WORD   ^M<R2, R3, R4>          ; Entry mask
                   1C  10  006E            BSBB    SETUP                   ; Set up R0:R3 and identify condition.
                           0070                                            ; Return only if FLTOVF or FLTDIV.
      50    01    0F   78  0070            ASHL    #15, #1, R0             ; Make the default result -0.0.
       54      00'8F   9A  0074            MOVZBL  #MTH$K_FLOOVEMAT, R4    ; Report MTH$_FLOOVEMAT, not SS$_FLTxxx.
                           0078
               10 A2   DD  0078    DO_SIG: PUSHL   SF$L_SAVE_PC(R2)        ; Report caller's PC, not exception PC.
                   54  DD  007B            PUSHL   R4                      ; Report MTH$_xxx, not SS$_xxx.
    00000000'GF   02   FB  007D            CALLS   #2, G^MTH$$SIGNAL       ; Signal the condition.
         0C A3    50   7D  0084    CON_U:  MOVQ    R0, CHF$L_MCH_SAVR0(R3) ; If continued, restore R0 and R1.
            50    00'  D0  0088            MOVL    S^#SS$_CONTINUE, R0     ; Continue from the original exception.
                       04  008B    DO_RET: RET                             ; Exit from handler.
```

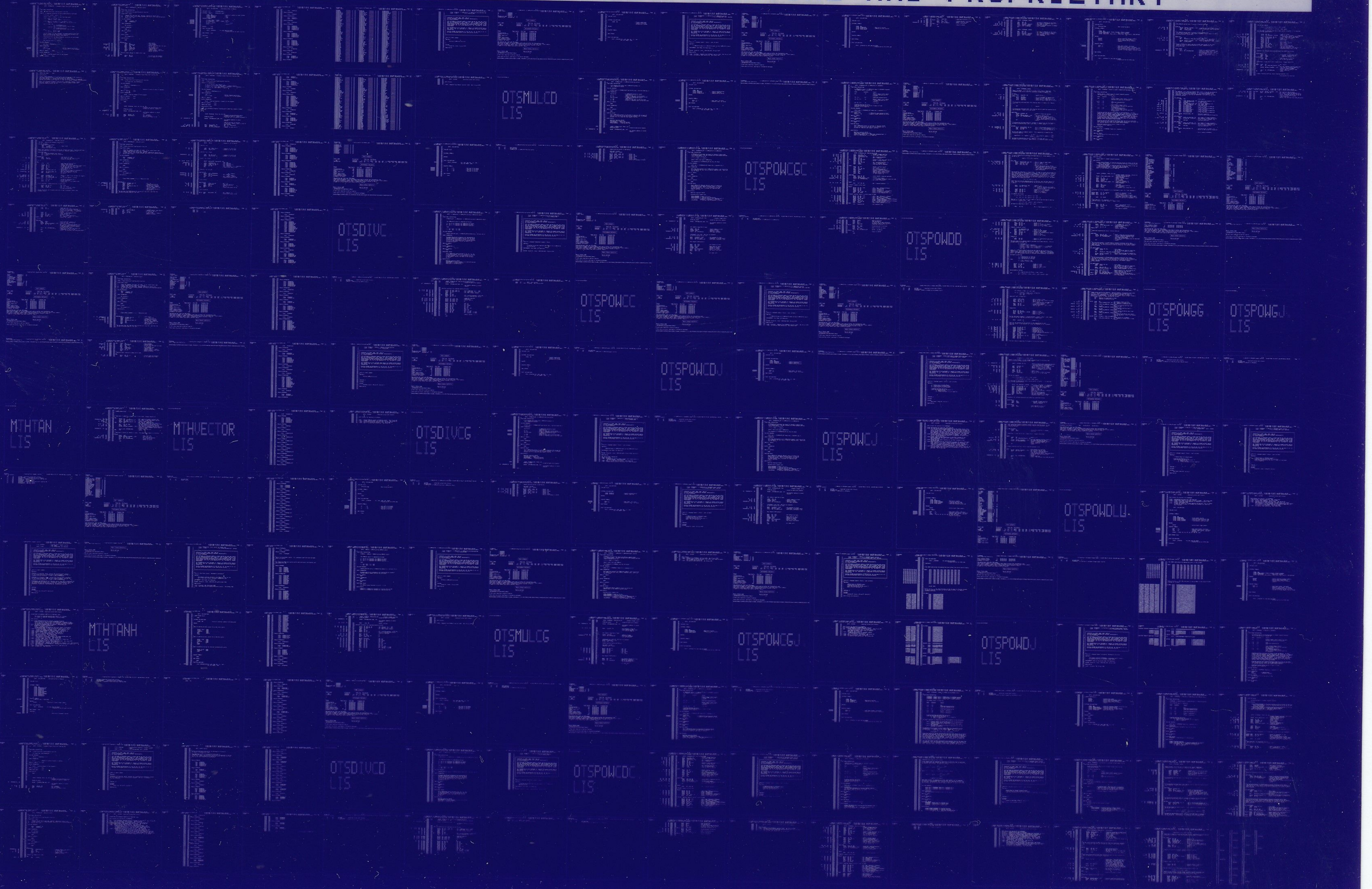
```
                        008C
                        008C   ;+
                        008C   ; Common setup routine for handlers.  Returns normally if exception was FLTOVF,
                        008C   ; FLTOVF_F, FLTDIV, or FLTDIV_F.  If the exception was anything else, it
                        008C   ; executes a RET, causing an exit from the handler with R0 = 0, which is
                        008C   ; equivalent to SS$_RESIGNAL.  In the case of a normal return (FLTOVF, FLTOVF_F,
                        008C   ; FLTDIV, or FLTDIV_F) it sets up R0:R3 as follows:
                        008C   ;          R0/R1:   0
                        008C   ;          R2:      address of establisher's frame
                        008C   ;          R3:      address of mechanism array
                        008C   ;-
                        008C
             50     7C  008C   SETUP:  CLRQ     R0                          ; Set default result to 0.0.
   52     04 AC     7D  008E           MOVQ     CHF$L_SIGARGLST(AP), R2     ; R2 = address of signal array
                        0092                                                ; R3 = address of mechanism array
0000'8F   04 A2     B1  0092           CMPW     CHF$L_SIG_NAME(R2), #SS$_FLTOVF
                        0098                                                ; Was it a floating overflow trap?
             18     13  0098           BEQL     DO_RSB                      ; Branch if yes.
0000'8F   04 A2     B1  009A           CMPW     CHF$L_SIG_NAME(R2), #SS$_FLTOVF_F
                        00A0                                                ; Or a floating overflow fault?
             10     13  00A0           BEQL     DO_RSB                      ; Branch if yes.
0000'8F   04 A2     B1  00A2           CMPW     CHF$L_SIG_NAME(R2), #SS$_FLTDIV
                        00A8                                                ; Or a floating divide by zero trap?
             08     13  00A8           BEQL     DO_RSB                      ; Branch if yes.
0000'8F   04 A2     B1  00AA           CMPW     CHF$L_SIG_NAME(R2), #SS$_FLTDIV_F
                        00B0                                                ; Or a floating divide by zero fault?
             D9     12  00B0           BNEQ     DO_RET                      ; None of the above: return from handler
                        00B2                                                ; with R0 = 0.
   08 A2  97 AF     9E  00B2   DO_RSB: MOVAB    B^POWGJX, CHF$L_SIG_NAME+4(R2)
                        00B7                                                ; Change return PC to POWGJX.
   52     04 A3     D0  00B7           MOVL     CHF$L_MCH_FRAME(R3), R2     ; R2 = address of establisher's frame
                    05  00BB           RSB                                  ; Return.
                        00BC
                        00BC           .END
```

```
BASE            = 00000004
CHF$L_MCH_FRAME= 00000004
CHF$L_MCH_SAVR0= 0000000C
CHF$L_SIGARGLST= 00000004
CHF$L_SIG_NAME = 00000004
CON_U             00000084 R       02
DO_RET            0000008B R       02
DO_RSB            00000082 R       02
DO_SIG            00000078 R       02
EXC_HNDLR_OVER    0000006C R       02
EXC_HNDLR_UNDER   0000005D R       02
EXP             = 0000000C
EXPGTR            00000023 R       02
MTH$$SIGNAL       ********    X    00
MTH$K_FLOOVEMAT   ********    X    00
MTH$K_FLOUNDMAT   ********    X    00
MTH$K_UNDEXP      ********    X    00
OTS$POWGJ         00000000 RG      02
PARTIAL           00000033 R       02
POWGJX            0000004C R       02
PSL$V_FU        = 00000006
SETUP             0000008C R       02
SF$L_SAVE_PC    = 00000010
SF$W_SAVE_PSW   = 00000004
SQUAR             00000027 R       02
SQUAR1            0000002C R       02
SS$_CONTINUE      ********    X    00
SS$_FLTDIV        ********    X    00
SS$_FLTDIV_F      ********    X    00
SS$_FLTOVF        ********    X    00
SS$_FLTOVF_F      ********    X    00
UNDEFINED         0000004D 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                         000000BC  (   188.)  02 (  2.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD   NOWRT NOVEC LONG

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

Phase                      Page faults    CPU Time       Elapsed Time
-----                      -----------    --------       ------------
Initialization                     35     00:00:00.09    00:00:00.97
Command processing                128     00:00:00.73    00:00:03.98
Pass 1                            137     00:00:02.04    00:00:08.13
Symbol table sort                   0     00:00:00.09    00:00:00.16
Pass 2                             68     00:00:00.92    00:00:03.90
Symbol table output                 4     00:00:00.03    00:00:00.04
Psect synopsis output               2     00:00:00.02    00:00:00.02
Cross-reference output              0     00:00:00.00    00:00:00.00
```

```
Assembler run totals             376      00:00:03.94    00:00:17.21

The working set limit was 1050 pages.
9091 bytes (18 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 106 non-local and 0 local symbols.
377 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
------------------                             --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     6

148 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

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

0264 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0265 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

OTSPOWHH
LIS

OTSPOWII
LIS

OTSPOWRJ
LIS

UVXPOWGG
LIS

OTSPOWGLU
LIS

OTSPOWHLU
LIS

OTSPOWHJ
LIS

UVXPOWCJ
LIS

UVXPOWRR
LIS

OTSPOWLUL
LIS

OTSPOWRR
LIS

OTSPOWJJ
LIS

UVXEXP
LIS

UVXGSINCO
LIS

OTSPOWRLU
LIS