```
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
```

**FILE**ID**OTSPOWCGJ

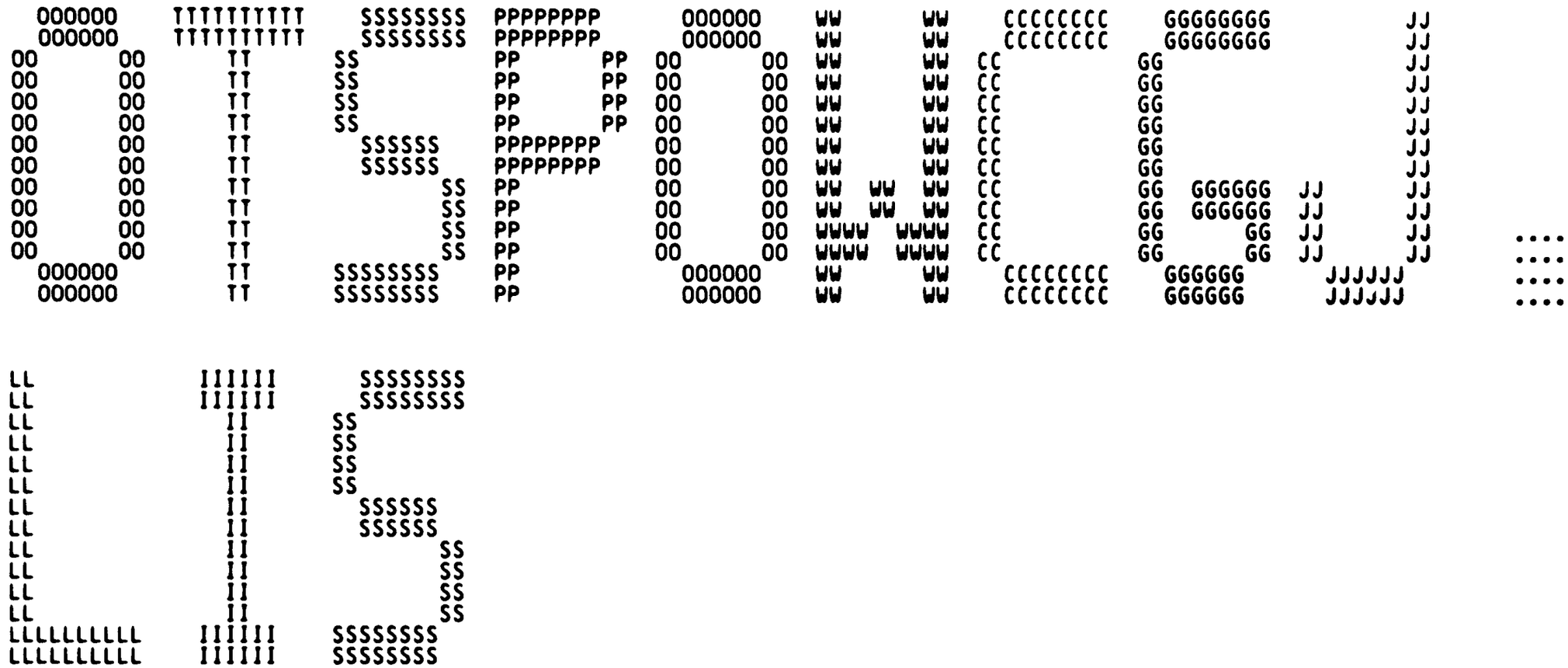

Table of contents

(2) 47 HISTORY ; Detailed Current Edit History
(3) 56 DECLARATIONS
(4) 90 OTS$POWCGJ_R3 - G COMPLEX*16 ** INTEGER*4

```
0000     1          .TITLE  OTS$POWCGJ - G COMPLEX*16 ** INTEGER*4 power routine
0000     2          .IDENT  /1-003/         ; File OTSPOWCGJ.MAR  Edit: SBL1003
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26 ;
0000    27 ;
0000    28 ;
0000    29 ; FACILITY: Language support library - user callable
0000    30 ;++
0000    31 ; ABSTRACT:
0000    32 ;
0000    33 ;       G COMPLEX*16 base to INTEGER*4 power.
0000    34 ;       Floating overflow can occur.
0000    35 ;       Undefined exponentiation can occur if
0000    36 ;       base = (0.,0.) and exp <=0
0000    37 ;
0000    38 ;--
0000    39 ;
0000    40 ; VERSION: 1
0000    41 ;
0000    42 ; HISTORY:
0000    43 ; AUTHOR:
0000    44 ;       Steven B. Lionel, 27-July-1979
0000    45 ;
```

```
0000     47          .SBTTL  HISTORY         ; Detailed Current Edit History
0000     48
0000     49
0000     50 ; Edit History
0000     51 ; 1-001 - Adapted from OTS$POWCJ version 1-003.  SBL 27-July-1979
0000     52 ; 1-002 - Fix bug in test for undefined exponentiation with negative powers.
0000     53 ;         SPR 11-35362  SBL 22-Jan-1981
0000     54 ; 1-003 - Use general mode addressing.  SBL 30-Nov-1981
```

```
                     .SBTTL  DECLARATIONS

                     ;
                     ; INCLUDE FILES:
                     ;

                     ; EXTERNAL SYMBOLS:
                     ;

                     .DSABL  GBL
                     .EXTRN  MTH$$SIGNAL             ; Math error routine
                     .EXTRN  OTS$DIVCG_R3            ; COMPLEX division routine
                     .EXTRN  MTH$K_UNDEXP

                     ;
                     ; MACROS:
                     ;

                     ;
                     ; EQUATED SYMBOLS:
                     ;

                     ;
                     ; OWN STORAGE:
                     ;

                     ;
                     ; PSECT DECLARATIONS:
                     ;

                     .PSECT  _OTS$CODE PIC,SHR,LONG,EXE,NOWRT
                                                     ; program section for OTS$ code
```

```
                     0000    90          .SBTTL  OTS$POWCGJ_R3 - G COMPLEX*16 ** INTEGER*4
                     0000    91 ;++
                     0000    92 ; FUNCTIONAL DESCRIPTION:
                     0000    93 ;
                     0000    94 ;       G COMPLEX*16 result = G COMPLEX*16 base ** INTEGER*4 exponent
                     0000    95 ;       The COMPLEX result is given by:
                     0000    96 ;
                     0000    97 ;       base            exponent        result
                     0000    98 ;
                     0000    99 ;       any               >0            PRODUCT (base * 2**i) where
                     0000   100 ;                                       i is each non-zero bit in
                     0000   101 ;                                       exponent.
                     0000   102 ;
                     0000   103 ;       (0., 0.)          <=0           Undefined exponentiation.
                     0000   104 ;
                     0000   105 ;       not (0., 0.)      <0            PRODUCT (base * 2**i) where
                     0000   106 ;                                       i is each non-zero bit in
                     0000   107 ;                                       !exponent!.
                     0000   108 ;
                     0000   109 ;       not (0., 0.)      =0            (1.0, 0.0)
                     0000   110 ;
                     0000   111 ;       Floating overflow can occur.
                     0000   112 ;       Undefined exponentiation occurs if base is 0 and
                     0000   113 ;       exponent is 0 or negative.
                     0000   114 ;
                     0000   115 ; CALLING SEQUENCE:
                     0000   116 ;
                     0000   117 ;       result.wgc.v = OTS$POWCGJ_R3 (base.rgc.v, exponent.rl.v)
                     0000   118 ;
                     0000   119 ; INPUT PARAMETERS:
00000004             0000   120 ;       base    = 4             ; G COMPLEX*16 base passed by VALUE'
00000014             0000   121 ;       exponent = 20           ; Longword integer exponent by value.
                     0000   122 ;
                     0000   123 ; IMPLICIT INPUTS:
                     0000   124 ;       NONE
                     0000   125 ;
                     0000   126 ; OUTPUT PARAMETERS:
                     0000   127 ;       NONE
                     0000   128 ;
                     0000   129 ; IMPLICIT OUTPUTS:
                     0000   130 ;       NONE
                     0000   131 ;
                     0000   132 ; FUNCTION VALUE:
                     0000   133 ;
                     0000   134 ;       THE G COMPLEX*16 result is returned in registers R0-R3.
                     0000   135 ;       This is a violation of the VAX calling standard, but is
                     0000   136 ;       excused for compiled code support routines.
                     0000   137 ;
                     0000   138 ;
                     0000   139 ; SIDE EFFECTS:
                     0000   140 ;
                     0000   141 ;       Modifies registers R0-R3!
                     0000   142 ;       SS$_FLTOVF - Floating overflow
                     0000   143 ;       SIGNALs MTH$_UNDEXP (82 = ' UNDEFINED EXPONENTATION') if
                     0000   144 ;       base is 0 and exponent is 0 or negative.
                     0000   145 ;
                     0000   146 ;--
```

```
                        01F0  0000   148          .ENTRY  OTS$POWCGJ_R3, ^M<R4,R5,R6,R7,R8>
                              0002   149                                  ; disable integer overflow
          54    04 AC    7D  0002   150          MOVQ    base(AP), R4         ; R4-R7 gets COMPLEX base
          56    0C AC    7D  0006   151          MOVQ    base+8(AP), R6
          58    14 AC    D0  000A   152          MOVL    exponent(AP), R8     ; R8 = longword exponent
                   03    18  000E   153          BGEQ    1$                   ; R8 = ! exponent !
             58    58    CE  0010   154          MNEGL   R8, R8
          11 58    00    E5  0013   155 1$:      BBCC    #0, R8, EVEN         ; branch if even and clear low bit
             50    54 50FD  0017   156          MOVG    R4, R0               ; R0-R3 = initial result
             52    56 50FD  001B   157          MOVG    R6, R2
       58    58    FF 8F    9C  001F   158          ROTL    #-1, R8, R8          ; R8 = unsigned_exponent / 2
                   6C    13  0024   159          BEQL    DONE                 ; done if exponent was 1
                   30    11  0026   160          BRB     SQUAR1               ; else use rest of exponent
                             0028   161
                             0028   162 EVEN:
             50    08 50FD  0028   163          MOVG    #1, R0               ; R0-R3 = initial result
                   52    7C  002C   164          CLRQ    R2                   ; (1.0, 0.0)
       58    58    FF 8F    9C  002E   165          ROTL    #-1, R8, R8          ; R8 = unsigned_exponent / 2
                   23    12  0033   166          BNEQ    SQUAR1               ; branch if exponent not 0
                   54 53FD  0035   167          TSTG    R4                   ; exponent was 0, test RP(base)
                   58    12  0038   168          BNEQ    DONE                 ; done if non-0, answer is 1.0
                   56 53FD  003A   169          TSTG    R6                   ; IP(base) better not be zero
                   53    12  003D   170          BNEQ    DONE                 ; it isn't return 1.0
                             003F   171
                             003F   172 UNDEFINED:
          50    01    0F    79  003F   173          ASHQ    #15, #1, R0          ; return R0-R3 = reserved operands
          52    01    0F    79  0043   174          ASHQ    #15, #1, R2
            7E    00'8F    9A  0047   175          MOVZBL  #MTH$K_UNDEXP, -(SP) ; FORTRAN error number
00000000'GF    01    FB  004B   176          CALLS   #1, G^MTH$$SIGNAL            ; convert to 32-bit condition code
                             0052   177                                       ; and SIGNAL MTH$_UNDEXP
                        04   0052   178          RET
                             0053   179
                             0053   180 SQUAR:
       58    58    FF 8F    78  0053   181          ASHL    #-1, R8, R8          ; R8 = !reduced exponent! / 2
                             0058   182 ;
                             0058   183 ; R4-R7 = square current base
                             0058   184 ;
                             0058   185 SQUAR1:
          7E    56    54 45FD  0058   186          MULG3   R4, R6, -(SP)        ; (SP) = tmp = RP(base)*IP(base)
                54    54 44FD  005D   187          MULG2   R4, R4               ; R4-R5 = RP(base)**2
                56    56 44FD  0061   188          MULG2   R6, R6               ; R6-R7 = IP(base)**2
                54    56 42FD  0065   189          SUBG2   R6, R4               ; R4-R5 = RP(base)**2 - IP(base)**2
          56    8E    6E 41FD  0069   190          ADDG3   (SP), (SP)+, R6      ; R6-R7 = 2*(RP(base)*IP(base))
                  E2 58    E9  006E   191          BLBC    R8, SQUAR            ; branch if next exponent bit is 0
                             0071   192 ;
                             0071   193 ; R0-R3 = partial result * current power of base
                             0071   194 ;
          7E    56    50 45FD  0071   195          MULG3   R0, R6, -(SP)        ; (SP) = tmp = RP(part) * IP(base)
                50    54 44FD  0076   196          MULG2   R4, R0               ; R0-R1 = RP(part) * RP(base)
          7E    56    52 45FD  007A   197          MULG3   R2, R6, -(SP)        ; (SP) = tmp = IP(part) * IP(base)
                50    8E 42FD  007F   198          SUBG2   (SP)+, R0            ; R0-R1 = RP(part)*RP(base)-IP(part)*IP(base
                52    54 44FD  0083   199          MULG2   R4, R2               ; R2-R3 = IP(part)*RP(base)
                52    8E 40FD  0087   200          ADDG2   (SP)+, R2            ; R2-R3 = IP(part)*RP(base)+RP(part)*IP(base
       58    58    FF 8F    78  008B   201          ASHL    #-1, R8, R8          ; R8 = !reduced exponent! / 2
                   C6    12  0090   202          BNEQ    SQUAR1               ; loop if more exponent bits left
                             0092   203 DONE:
                14 AC    D5  0092   204          TSTL    exponent(AP)         ; test exponent sign
```

```
                     1D   18  0095   205          BGEQ    POWCGJ                      ; done if positive
                     50 53FD  0097   206          TSTG    R0                          ; test RP(result)
                     05   12  009A   207          BNEQ    RECIP                       ; if non-0, OK to take reciprocal
                     52 53FD  009C   208          TSTG    R2                          ; RP(result) was 0, test IP(result)
                     9E   13  009F   209          BEQL    UNDEFINED                   ; undefined (0.0+0.0i) ** -n
                              00A1   210 RECIP:
                  7E 52   7D  00A1   211          MOVQ    R2, -(SP)                   ; second arg pair is divisor
                  7E 50   7D  00A4   212          MOVQ    R0, -(SP)
                  7E 7C       00A7   213          CLRQ    -(SP)                       ; push (1.0,0.0) on stack
                  7E 08 50FD  00A9   214          MOVG    #1, -(SP)
         00000000'GF 08   FB  00AD   215          CALLS   #8, G^OTS$DIVCG_R3          ; R0-R3 = reciprocal
                              00B4   216 POWCGJ:
                     04       00B4   217          RET                                 ; result in R0-R3
                              00B5   218
                              00B5   219          .END
```

```
BASE                = 00000004
DONE                  00000092 R      01
EVEN                  00000028 R      01
EXPONENT            = 00000014
MTH$$SIGNAL           ********   X    00
MTH$K_UNDEXP          ********   X    00
OTS$DIVCG_R3          ********   X    00
OTS$POWCGJ_R3         00000000 RG     01
POWCGJ                000000B4 R      01
RECIP                 000000A1 R      01
SQUAR                 00000053 R      01
SQUAR1                00000058 R      01
UNDEFINED             0000003F R      01
```

## Psect synopsis

| PSECT name | Allocation | PSECT No. | Attributes |
|---|---|---|---|
| . ABS . | 00000000 ( 0.) | 00 ( 0.) | NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE |
| _OTS$CODE | 000000B5 ( 181.) | 01 ( 1.) | PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG |

## Performance indicators

| Phase | Page faults | CPU Time | Elapsed Time |
|---|---|---|---|
| Initialization | 30 | 00:00:00.10 | 00:00:01.02 |
| Command processing | 108 | 00:00:00.48 | 00:00:02.73 |
| Pass 1 | 75 | 00:00:00.65 | 00:00:02.91 |
| Symbol table sort | 0 | 00:00:00.01 | 00:00:00.01 |
| Pass 2 | 52 | 00:00:00.51 | 00:00:02.52 |
| Symbol table output | 2 | 00:00:00.01 | 00:00:00.41 |
| Psect synopsis output | 2 | 00:00:00.03 | 00:00:00.06 |
| Cross-reference output | 0 | 00:00:00.00 | 00:00:00.00 |
| Assembler run totals | 271 | 00:00:01.81 | 00:00:09.75 |

The working set limit was 900 pages.
3177 bytes (7 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 13 non-local and 1 local symbols.
219 source lines were read in Pass 1, producing 11 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

## Macro library statistics

| Macro library name | Macros defined |
|---|---|
| _$255$DUA28:[SYSLIB]STARLET.MLB;2 | 0 |

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

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

OTSMULCD
LIS

OTSPOWCGC
LIS

OTSDIVC
LIS

OTSPOWDD
LIS

OTSPOWCC
LIS

OTSPOWGG
LIS

OTSPOWGJ
LIS

OTSPOWCDJ
LIS

MTHTAN
LIS

MTHVECTOR
LIS

OTSDIVCG
LIS

OTSPOWCJ
LIS

OTSPOWDLJ
LIS

MTHTANH
LIS

OTSMULCG
LIS

OTSPOWCGJ
LIS

OTSPOWDJ
LIS

OTSDIVCD
LIS

OTSPOWCDC
LIS