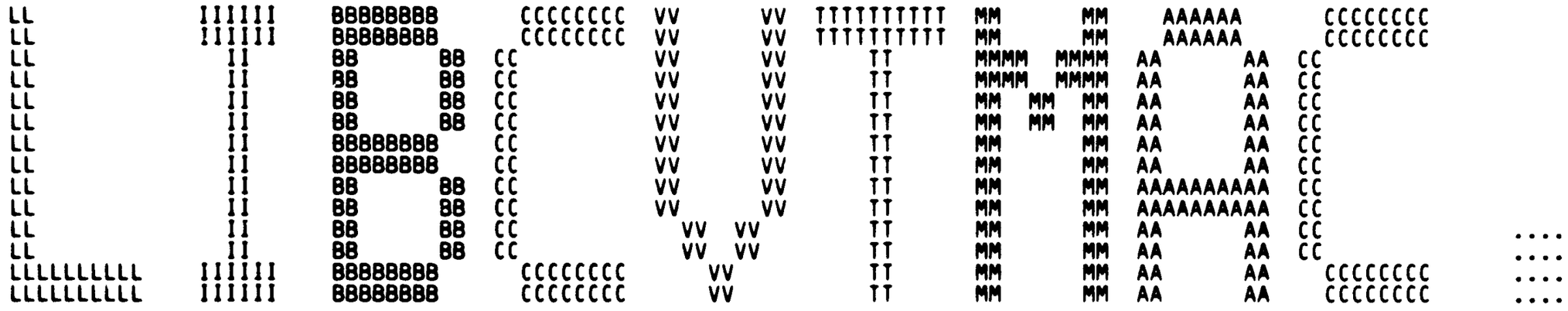
```
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                 IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBB         BBB  RRR         RRR        TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBBBBBBBBBBB     RRRRRRRRRRRR           TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR   RRR              TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLL                    III        BBB         BBB  RRR      RRR           TTT        LLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**LIBCVTMAC



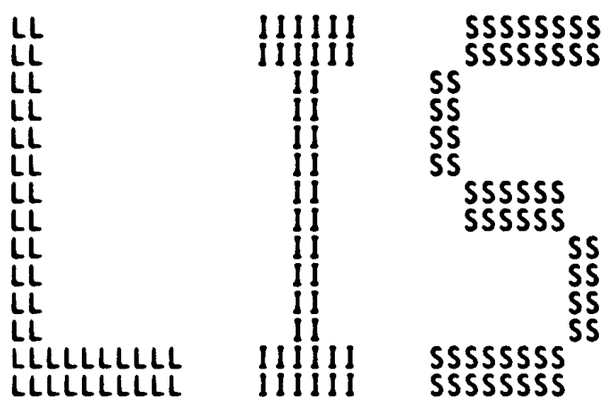


Table of contents

```
0000     1          .TITLE  LIB$CVTMAC      MACRO-32 support for LIB$CVT_DX_DX
0000     2          .IDENT  /1-005/         ; File: LIBCVTMAC.MAR  EDIT: PDG1005
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  ; FACILITY: Run-time library.
0000    30  ;
0000    31  ; ABSTRACT:
0000    32  ;       This module contains the MACRO-32 support routines for LIB$CVT_DX_DX
0000    33  ;       routine.  All of the entry points are JSB entry points.
0000    34  ;
0000    35  ; ENVIRONMENT:
0000    36  ;       Some entry points require the Caller to have LIB$EMULATE or the
0000    37  ;       machine have the full instruction set of VAX-11 architecture.
0000    38  ;       Note that this module is for specific support of LIB$CVT_DX_DX routine,
0000    39  ;       hence it has no  RTL "globally defined" entry point, therefore this
0000    40  ;       module assumes that the caller has IV, FU, DV bits in PSW turned on.
0000    41  ;
0000    42  ; AUTHOR:
0000    43  ;       AUTHORS :  FAROKH MORSHED, PETER GILBERT  7-FEB-81, VERSION 1, VMS V3.0.
0000    44  ;
0000    45  ; MODIFIED BY:
0000    46  ;
0000    47  ; VERSION: 1
0000    48  ; 1001  Original.
0000    49  ; 1002  Make references to LIB$SIGNAL General mode addressing.  FM 30-OCT-81.
0000    50  ; 1003  Fix LIB$$CVT_CVTRHQ_R1 to round correctly.  FM 1-Mar-83
0000    51  ; 1004  Fix the bug that a zero integer to float does not look for a zero
0000    52  ;       value.  FM 13-Jan-84
0000    53  ; 1005  Fix bug in LIB$$CVT_CVTRHO_R1 in code avoiding integer overflow.
0000    54  ;       PDG 10-Jul-1984
0000    55  ;--
0000    56  ;
```

```
                     58          .SBTTL  DECLARATIONS
                     59
                     60 ;
                     61 ; PSECT DECLARATIONS:
                     62 ;
00000000             63          .PSECT  _LIB$CODE       PIC, SHR, LONG, EXE, NOWRT
                     64 ;
                     65 ; EXTERNAL ROUTINES
                     66 ;
                     67          .EXTERNAL       LIB$SIGNAL
                     68 ;
                     69 ;       MACRO
                     70 ;
                     71          $SSDEF
                     72
```

```
                     0000     74          .SBTTL  LIB$$CVT_CVTLB_R1
                     0000     75
                     0000     76 ;++
                     0000     77 ; FUNCTIONAL DESCRIPTION:
                     0000     78 ;       Convert LONGWORD to BYTE
                     0000     79 ;
                     0000     80 ;
                     0000     81 ; CALLING SEQUENCE:
                     0000     82 ;       LIB$$CVT_CVTLB_R1 ( long.rl.r, byte.wb.r )
                     0000     83 ;       This is a JSB entry point.
                     0000     84 ;
                     0000     85 ;
                     0000     86 ; FORMAL PARAMETERS:
                     0000     87 ;       R0 --> long     R1 --> byte
                     0000     88 ;
                     0000     89 ;
                     0000     90 ; IMPLICIT INPUTS:
                     0000     91 ;       NONE
                     0000     92 ;
                     0000     93 ;
                     0000     94 ; IMPLICIT OUTPUTS:
                     0000     95 ;       NONE
                     0000     96 ;
                     0000     97 ;
                     0000     98 ; COMPLETION CODES:
                     0000     99 ;       NONE
                     0000    100 ;
                     0000    101 ; SIDE EFFECTS:
                     0000    102 ;
                     0000    103 ;
                     0000    104 ;--
                     0000    105
                     0000    106 LIB$$CVT_CVTLB_R1 ::
 61    60    F6      0000    107         CVTLB   (R0), (R1)              ;Convert LONGWORD to BYTE
                     05 0003    108         RSB
```

```
                          0004   110          .SBTTL  LIB$$CVT_CVTLW_R1
                          0004   111
                          0004   112 ;++
                          0004   113 ; FUNCTIONAL DESCRIPTION:
                          0004   114 ;       Convert LONGWORD to WORD.
                          0004   115 ;
                          0004   116 ;
                          0004   117 ; CALLING SEQUENCE:
                          0004   118 ;       LIB$$CVT_CVTLW_R1 ( long.rl.r, word.ww.r )
                          0004   119 ;       This is a JSB entry point.
                          0004   120 ;
                          0004   121 ;
                          0004   122 ; FORMAL PARAMETERS:
                          0004   123 ;       R0 --> long     R1 --> word
                          0004   124 ;
                          0004   125 ;
                          0004   126 ; IMPLICIT INPUTS:
                          0004   127 ;       NONE
                          0004   128 ;
                          0004   129 ;
                          0004   130 ; IMPLICIT OUTPUTS:
                          0004   131 ;       NONE
                          0004   132 ;
                          0004   133 ;
                          0004   134 ; COMPLETION CODES:
                          0004   135 ;       NONE
                          0004   136 ;
                          0004   137 ; SIDE EFFECTS:
                          0004   138 ;
                          0004   139 ;
                          0004   140 ;--
                          0004   141
                          0004   142 LIB$$CVT_CVTLW_R1 ::
 61    60    F7   05      0004   143          CVTLW   (R0), (R1)              ;Convert LONGWORD to WORD
                   05      0007   144          RSB
```

```
                          0008   146          .SBTTL  LIB$$CVT_CVTLH_R1
                          0008   147
                          0008   148 ;++
                          0008   149 ; FUNCTIONAL DESCRIPTION:
                          0008   150 ;       Convert LONGWORD to H_FLOATING.
                          0008   151 ;
                          0008   152 ;
                          0008   153 ; CALLING SEQUENCE:
                          0008   154 ;       LIB$$CVT_CVTLH_R1 ( long.rl.r, hfloat.wh.r )
                          0008   155 ;       This is a JSB entry point.
                          0008   156 ;
                          0008   157 ;
                          0008   158 ; FORMAL PARAMETERS:
                          0008   159 ;       R0 --> long     R1 --> hfloat
                          0008   160 ;
                          0008   161 ;
                          0008   162 ; IMPLICIT INPUTS:
                          0008   163 ;       NONE
                          0008   164 ;
                          0008   165 ;
                          0008   166 ; IMPLICIT OUTPUTS:
                          0008   167 ;       NONE
                          0008   168 ;
                          0008   169 ;
                          0008   170 ; COMPLETION CODES:
                          0008   171 ;       NONE
                          0008   172 ;
                          0008   173 ; SIDE EFFECTS:
                          0008   174 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                          0008   175 ;
                          0008   176 ;--
                          0008   177
                          0008   178 LIB$$CVT_CVTLH_R1 ::
61    60 6EFD             0008   179          CVTLH   (R0), (R1)              ;Convert LONGWORD to H_FLOATING
                    05    000C   180          RSB
                          000D   181
```

```
                                      000D   183          .SBTTL  LIB$$CVT_CVTROUF_R1
                                      000D   184
                                      000D   185 ;++
                                      000D   186 ; FUNCTIONAL DESCRIPTION:
                                      000D   187 ;       Convert unsigned OCTAWORD to FLOAT.
                                      000D   188 ;
                                      000D   189 ;
                                      000D   190 ; CALLING SEQUENCE:
                                      000D   191 ;       LIB$$CVT_CVTROUF_R1 ( octa.ro.r, float.wf.r )
                                      000D   192 ;       This is a JSB entry point.
                                      000D   193 ;
                                      000D   194 ;
                                      000D   195 ; FORMAL PARAMETERS:
                                      000D   196 ;       R0 --> octa     R1 --> float
                                      000D   197 ;
                                      000D   198 ;
                                      000D   199 ; IMPLICIT INPUTS:
                                      000D   200 ;       NONE
                                      000D   201 ;
                                      000D   202 ;
                                      000D   203 ; IMPLICIT OUTPUTS:
                                      000D   204 ;       NONE
                                      000D   205 ;
                                      000D   206 ;
                                      000D   207 ; COMPLETION CODES:
                                      000D   208 ;       NONE
                                      000D   209 ;
                                      000D   210 ; SIDE EFFECTS:
                                      000D   211 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                                      000D   212 ;
                                      000D   213 ;--
                                      000D   214
                                      000D   215 ;
                                      000D   216 ;         DEFINE SOME CONSTANTS
                                      000D   217 ;
                00000011              000D   218 FDUMMY: .BLKL           1
                00007080              0011   219          .F_FLOATING    79228162514264337593543950336   ;2**96
                00006080              0015   220          .F_FLOATING    18446744073709551616            ;2**64
                00005080              0019   221 F32:     .F_FLOATING    4294967296                      ;2**32
                00004080              001D   222          .F_FLOATING    1                               ;
                                      0021   223 ;
                                      0021   224 ;         FIND THE FLOATING ATOMIC DATA TYPE BY MULTIPLYING EACH LONGWORD
                                      0021   225 ;         OF OCTAWORD BY AN APPROPRIATE CONSTANT.
                                      0021   226 ;
                                      0021   227 LIB$$CVT_CVTROUF_R1 ::
                     0F   BB          0021   228          PUSHR   #^M<R0,R1,R2,R3>
                     61   D4          0023   229          CLRF    (R1)                    ;Initialize destination.
                  52 04   D0          0025   230          MOVL    #4, R2                  ;Set up the loop counter for 4 times
                  53 60   4E          0028   231 10$:     CVTLF   (R0), R3                ;Convert the next LONGWORD to floating
                     06   14          002B   232          BGTR    13$                     ;This longword is positive
                     0C   13          002D   233          BEQL    15$                     ;If zero, then just skip it
               53 E7 AF   40          002F   234          ADDF2   F32, R3                 ;Negative, so add the difference
            53 D6 AF42    44          0033   235 13$:     MULF2   FDUMMY[R2], R3          ;Multiply the result by the constant
                  61 53   40          0038   236          ADDF2   R3, (R1)                ;Add in the result to destination
                     52   97          003B   237 15$:     DECB    R2                      ;One less time to go around in the loop
                     08   13          003D   238          BEQL    20$                     ;Finished if counter is zero
                     80   D5          003F   239          TSTL    (R0)+                   ;Next longword of OCTA
```

```
60   D5  0041   240          TSTL    (R0)                    ;Is it zero ?
F6   13  0043   241          BEQL    15$                     ;Yes, so ignore it
E1   11  0045   242          BRB     10$                     ;Loop to chomp some more on the OCTA
             0047   243 20$:
0F   BA  0047   244          POPR    #^M<R0,R1,R2,R3>
     05  0049   245          RSB
```

```
                                004A   247          .SBTTL  LIB$$CVT_CVTROUD_R1
                                004A   248
                                004A   249 ;++
                                004A   250 ; FUNCTIONAL DESCRIPTION:
                                004A   251 ;       Convert unsigned OCTAWORD to D_FLOATING.
                                004A   252 ;
                                004A   253 ;
                                004A   254 ; CALLING SEQUENCE:
                                004A   255 ;       LIB$$CVT_CVTROUD_R1 ( octa.ro.r, dfloat.wd.r )
                                004A   256 ;       This is a JSB entry point.
                                004A   257 ;
                                004A   258 ;
                                004A   259 ; FORMAL PARAMETERS:
                                004A   260 ;       R0 --> octa     R1 --> dfloat
                                004A   261 ;
                                004A   262 ;
                                004A   263 ; IMPLICIT INPUTS:
                                004A   264 ;       NONE
                                004A   265 ;
                                004A   266 ;
                                004A   267 ; IMPLICIT OUTPUTS:
                                004A   268 ;       NONE
                                004A   269 ;
                                004A   270 ;
                                004A   271 ; COMPLETION CODES:
                                004A   272 ;       NONE
                                004A   273 ;
                                004A   274 ; SIDE EFFECTS:
                                004A   275 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                                004A   276 ;
                                004A   277 ;--
                                004A   278
                                004A   279 ;
                                004A   280 ;         DEFINE SOME CONSTANTS
                                004A   281 ;
                    00000052    004A   282 DDUMMY:  .BLKL           2
00000000 00007080               0052   283          .D_FLOATING     79228162514264337593543950336   ;2**96
00000000 00006080               005A   284          .D_FLOATING     18446744073709551616            ;2**64
00000000 00005080               0062   285 D32:     .D_FLOATING     4294967296                      ;2**32
00000000 00004080               006A   286          .D_FLOATING     1                               ;
                                0072   287 ;
                                0072   288 ;         FIND THE FLOATING ATOMIC DATA TYPE BY MULTIPLYING EACH LONGWORD
                                0072   289 ;         OF OCTAWORD BY AN APPROPRIATE CONSTANT.
                                0072   290 ;
                                0072   291 LIB$$CVT_CVTROUD_R1 ::
                      1F    BB  0072   292          PUSHR   #^M<R0,R1,R2,R3,R4>
                      61    7C  0074   293          CLRD    (R1)                    ;Initialize destination
                52    04    D0  0076   294          MOVL    #4, R2                  ;Set up the loop counter for 4 times
                53    60    6E  0079   295 10$:     CVTLD   (R0), R3                ;Convert the next LONGWORD to floating
                      06    14  007C   296          BGTR    13$                     ;This longword is positive
                      0C    13  007E   297          BEQL    15$                     ;If zero, then just skip it
          53    DF    AF    60  0080   298          ADDD2   D32, R3                 ;Negative, so add the difference
     53   C2    AF42        64  0084   299 13$:     MULD2   DDUMMY[R2], R3          ;Multiply the result by the constant
                61    53    60  0089   300          ADDD2   R3, (R1)                ;Add in the result to destination
                      52    97  008C   301 15$:     DECB    R2                      ;One less time to go around in the loop
                      08    13  008E   302          BEQL    20$                     ;Finished if counter is zero
                      80    D5  0090   303          TSTL    (R0)+                   ;Next longword of OCTA
```

```
60   D5  0092   304          TSTL    (R0)                    ;Is it zero
F6   13  0094   305          BEQL    15$                     ;Yes, so ignore it
E1   11  0096   306          BRB     10$                     ;Loop to chomp some more on the OCTA
             0098   307 20$:
1F   BA  0098   308          POPR    #^M<R0,R1,R2,R3,R4>
     05  009A   309          RSB
```

```
                                                .SBTTL  LIB$$CVT_CVTROUG_R1

                                        ;++
                                        ; FUNCTIONAL DESCRIPTION:
                                        ;       Convert unsigned OCTAWORD to G_FLOATING
                                        ;
                                        ; CALLING SEQUENCE:
                                        ;       LIB$$CVT_CVTROUG_R1 ( octa.ro.r, gfloat.wg.r )
                                        ;       This is a JSB entry point.
                                        ;
                                        ; FORMAL PARAMETERS:
                                        ;       R0 --> octa     R1 --> gfloat
                                        ;
                                        ; IMPLICIT INPUTS:
                                        ;       NONE
                                        ;
                                        ; IMPLICIT OUTPUTS:
                                        ;       NONE
                                        ;
                                        ; COMPLETION CODES:
                                        ;       NONE
                                        ; SIDE EFFECTS:
                                        ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                                        ;--

                                        ;
                                        ;         DEFINE SOME CONSTANTS
                                        ;
                 000000A3               GDUMMY: .BLKL           2
00000000 00004610                               .G_FLOATING     79228162514264337593543950336;2**96
00000000 00004410                               .G_FLOATING     18446744073709551616;2**64
00000000 00004210                       G32:    .G_FLOATING     4294967296;2**32
00000000 00004010                               .G_FLOATING     1;
                                        ;
                                        ;        FIND THE FLOATING ATOMIC DATA TYPE BY MULTIPLYING EACH LONGWORD
                                        ;        OF OCTAWORD BY AN APPROPRIATE CONSTANT.
                                        ;
                                        LIB$$CVT_CVTROUG_R1 ::
                      1F   BB           PUSHR   #^M<R0,R1,R2,R3,R4>
                      61   7C           CLRG    (R1)                    ;Initialize destination
                52    04   D0           MOVL    #4, R2                  ;Set up the loop counter for 4 times
                53    60 4EFD           CVTLG   (R0)+, R3               ;Convert the next LONGWORD to floating
                      07   14           BGTR    13$                     ;This longword is positive
                      0F   13           BEQL    15$                     ;If zero, then just skip it
          53    DD AF 40FD              ADDG2   G32, R3                 ;Negative, so add the difference
53        BF AF42 44FD                  MULG2   GDUMMY[R2], R3          ;Multiply the result by the constant
                61    53 40FD           ADDG2   R3, (R1)                ;Add in the result to destination
                      52   97           DECB    R2                      ;One less time to go around in the loop
                      08   13           BEQL    20$                     ;Finished if counter is zero
                      80   D5           TSTL    (R0)+                   ;Next longword of OCTA
```

```
60   D5  00E7   368          TSTL    (R0)                        ;Is it zero ?
F6   13  00E9   369          BEQL    15$                         ;Yes, so ignore it
DD   11  00EB   370          BRB     10$                         ;Loop to chomp some more on the OCTA
         00ED   371 20$:
1F   BA  00ED   372          POPR    #^M<R0,R1,R2,R3,R4>
     05  00EF   373          RSB
```

```
                                00F0   375          .SBTTL  LIB$$CVT_CVTROUH_R1
                                00F0   376
                                00F0   377 ;++
                                00F0   378 ; FUNCTIONAL DESCRIPTION:
                                00F0   379 ;       Convert unsigned OCTAWORD to H_FLOATING rounded.
                                00F0   380 ;
                                00F0   381 ;
                                00F0   382 ; CALLING SEQUENCE:
                                00F0   383 ;       LIB$$CVT_CVTROUH_R1 ( octa.ro.r, hfloat.wh.r )
                                00F0   384 ;       This is a JSB entry point.
                                00F0   385 ;
                                00F0   386 ;
                                00F0   387 ; FORMAL PARAMETERS:
                                00F0   388 ;       R0 --> octa     R1 --> hfloat
                                00F0   389 ;
                                00F0   390 ;
                                00F0   391 ; IMPLICIT INPUTS:
                                00F0   392 ;       NONE
                                00F0   393 ;
                                00F0   394 ;
                                00F0   395 ; IMPLICIT OUTPUTS:
                                00F0   396 ;       NONE
                                00F0   397 ;
                                00F0   398 ;
                                00F0   399 ; COMPLETION CODES:
                                00F0   400 ;       NONE
                                00F0   401 ;
                                00F0   402 ; SIDE EFFECTS:
                                00F0   403 ;
                                00F0   404 ;--
                                00F0   405
                                00F0   406 LIB$$CVT_CVTROUH_R1 ::
                          0F    BB   00F0   407          PUSHR   #^M<R0,R1,R2,R3>
                          60    D5   00F2   408          TSTL    (R0)                    ;If OU is zero, return zero.
                          1A    12   00F4   409          BNEQ    5$                      ;....
                       04 A0    D5   00F6   410          TSTL    4(R0)                   ;....
                          15    12   00F9   411          BNEQ    5$                      ;....
                       08 A0    D5   00FB   412          TSTL    8(R0)                   ;....
                          10    12   00FE   413          BNEQ    5$                      ;....
                       0C A0    D5   0100   414          TSTL    12(R0)                  ;....
                          0B    12   0103   415          BNEQ    5$                      ;....
                          81    D4   0105   416          CLRL    (R1)+                   ;Return 0 in H.
                          81    D4   0107   417          CLRL    (R1)+                   ;....
                          81    D4   0109   418          CLRL    (R1)+                   ;....
                          61    D4   010B   419          CLRL    (R1)                    ;....
                        0079    31   010D   420          BRW     50$                     ;Go to RSB back to caller.
                                0110   421 5$:
                    5E    12    C2   0110   422          SUBL2   #18, SP                 ;Make room for nomalized OU.
                                0113   423 ;
                                0113   424 ; Move source to top of stack to normalize it
                                0113   425 ;
                    6E    60    7D   0113   426          MOVQ    (R0), (SP)              ;Put OU in here to be
            08 AE     08 A0    7D   0116   427          MOVQ    8(R0), 8(SP)            ;       worked on.
                      10 AE    B4   011B   428          CLRW    16(SP)                  ;We need this extra word for ADWC.
                                011E   429 ;
                                011E   430 ; Initialize some locations
                                011E   431 ;
```

```
                     52   D4  011E    432          CLRL    R2                      ;R2 will contain number of shifts.
                     53   D4  0120    433          CLRL    R3                      ;Temporary.
                     20   B9  0122    434          BICPSW  #^X20                   ;Turn off IV so we can ASHQ
                                 0124    435 ;
                                 0124    436 ; This loop will shift OU left until MSB is a one.
                                 0124    437 ;
                                 0124    438 10$:
                     52   D7  0124    439          DECL    R2                      ;Count number of shifts.
                   0063   30  0126    440          BSBW    SHIFT_OU_LEFT           ;Shift OU left one time.
                  0F AE   95  0129    441          TSTB    15(SP)                  ;Is MSB of normalized OU set ?
                     F6   18  012C    442          BGEQ    10$                     ;No, loop again.
                                 012E    443 ;
                                 012E    444 ; Round the OU if needed
                                 012E    445 ;
                1D 6E    0E   E1  012E    446          BBC     #14, (SP), 30$          ;If the bit beyond LSB of H clear no
                                 0132    447                                          ;       rounding is needed.
                19 6E    0F   E3  0132    448          BBCS    #15, (SP), 30$          ;If this bit is clear we just need
                                 0136    449                                          ;       to set it and it is rounded.
             01 AE    80 8F   8A  0136    450          BICB2   #^X80, 1(SP)            ;Painfully round this bit pattern.
                02 AE    01   C0  013B    451          ADDL2   #1, 2(SP)               ;.....
                06 AE    00   D8  013F    452          ADWC    #0, 6(SP)               ;.....
                0A AE    00   D8  0143    453          ADWC    #0, 10(SP)              ;.....
                0E AE    00   D8  0147    454          ADWC    #0, 14(SP)              ;.....
                                 014B    455 ;
                                 014B    456 ; Normalize this bit pattern
                                 014B    457 ;
                  03 10 AE   E8  014B    458          BLBS    16(SP), 40$             ;If this bit is on, it means that we
                                 014F    459                                          ;       had an overflow on the last
                                 014F    460                                          ;       word, so it is already
                                 014F    461                                          ;       normalized.
                                 014F    462 30$:
                     003A   30  014F    463          BSBW    SHIFT_OU_LEFT           ;Normalize it.
                                 0152    464 ;
                                 0152    465 ; Put what we have cooked up in the H destination
                                 0152    466 ;
                                 0152    467 40$:
          52   00000080 8F   C0  0152    468          ADDL2   #128, R2                ;exp = length of OU (128) - #of shifts.
                   61   52   B0  0159    469          MOVW    R2, (R1)                ;Put in exponent.
                01 A1    40 8F   88  015C    470          BISB2   #^X40, 1(R1)            ;Set the excess bit.
                02 A1    0E AE   B0  0161    471          MOVW    14(SP), 2(R1)           ;Put in the fraction.
                04 A1    0C AE   B0  0166    472          MOVW    12(SP), 4(R1)           ;.....
                06 A1    0A AE   B0  016B    473          MOVW    10(SP), 6(R1)           ;.....
                08 A1    08 AE   B0  0170    474          MOVW    8(SP), 8(R1)            ;.....
                0A A1    06 AE   B0  0175    475          MOVW    6(SP), 10(R1)           ;.....
                0C A1    04 AE   B0  017A    476          MOVW    4(SP), 12(R1)           ;.....
                0E A1    02 AE   B0  017F    477          MOVW    2(SP), 14(R1)           ;.....
                            20   B8  0184    478          BISPSW  #^X20                   ;Turn IV back on.
                         5E   12   C0  0186    479          ADDL2   #18, SP                 ;Restore SP
                            0F   BA  0189    480 50$:     POPR    #^M<R0,R1,R2,R3>
                            05   018B    481          RSB                             ;RSB back to caller.
                                 018C    482 ;
                                 018C    483 ; Subroutine to shift OU one to the left
                                 018C    484 ;
                                 018C    485 SHIFT_OU_LEFT:
                            53   94  018C    486          CLRB    R3                      ;Initialize R3
                         0B AE   95  018E    487          TSTB    11(SP)                  ;Is MSB of low quad set.
                            02   18  0191    488          BGEQ    10$                     ;No.
```

```
                          53   96  0193   489          INCB    R3                          ;Yes, remember it.
                                   0195   490 10$:
          04 AE    04 AE   01   79  0195   491          ASHQ    #1, 4(SP), 4(SP)            ;Shift low quad one time.
          0C AE    0C AE   01   79  019B   492          ASHQ    #1, 12(SP), 12(SP)          ;Shift high quad one time.
     0C AE    01    00     53   F0  01A1   493          INSV    R3, #0, #1, 12(SP)          ;If MSB of low quad was set then set
                                   01A7   494                                              ;LSB of high quad.
                                05  01A7   495          RSB                                 ;Return to main routine.
```

```
                                                .SBTTL  LIB$$CVT_CVTRDQ_R1

                                        ;++
                                        ; FUNCTIONAL DESCRIPTION:
                                        ;       Convert D_FLOATING to QUADWORD rounded.
                                        ;
                                        ;
                                        ; CALLING SEQUENCE:
                                        ;       LIB$$CVT_CVTRDQ_R1 ( dfloat.rd.r, quad.wq.r )
                                        ;       This is a JSB entry point.
                                        ;
                                        ;
                                        ; FORMAL PARAMETERS:
                                        ;       R0 --> dfloat   R1 --> QUADWORD
                                        ;
                                        ;
                                        ; IMPLICIT INPUTS:
                                        ;       NONE
                                        ;
                                        ;
                                        ; IMPLICIT OUTPUTS:
                                        ;       NONE
                                        ;
                                        ;
                                        ; COMPLETION CODES:
                                        ;       NONE
                                        ;
                                        ; SIDE EFFECTS:
                                        ;       NONE
                                        ;
                                        ;--

                        00000007        D_OFF   = 7                             ; Offset to exponent (double)

                                        LIB$$CVT_CVTRDQ_R1 ::
                0F   BB                 PUSHR   #^M<R0,R1,R2,R3>        ; Save these registers
                61   7C                 CLRD    (R1)                    ; Initialize result to zero
           52   60   70                 MOVD    (R0), R2                ; Grab source (reserved operand?)
                24   13                 BEQL    20$                     ; Branch if result is zero
     52    1000 8F   A2                 SUBW2   #32@D_OFF, R2           ; Scale down by 2**32
       04  A1   52   6B                 CVTRDL  R2, 4(R1)               ; Store high-order longword of result
     52    1000 8F   A0                 ADDW2   #32@D_OFF, R2           ; Scale up by 2**32
       7E    04 A1   6E                 CVTLD   4(R1), -(SP)            ; Convert long back to double
                05   13                 BEQL    10$                     ; Skip if zero
     6E    1000 8F   A0                 ADDW2   #32@D_OFF, (SP)         ; Scale up by 2**32
           52   8E   62                 SUBD2   (SP)+, R2               ; Subtract high-order longword
           61   52   6B                 CVTRDL  R2, (R1)                ; Convert rounded remainder to long
                03   18                 BGEQ    20$                     ; Skip if positive
             04 A1   D7                 DECL    4(R1)                   ; Sign-extend low-order longword
                0F   BA                 POPR    #^M<R0,R1,R2,R3>        ; Restore these registers
                     05                 RSB
```

```
                     01D8   549          .SBTTL  LIB$$CVT_CVTDH_R1
                     01D8   550
                     01D8   551 ;++
                     01D8   552 ; FUNCTIONAL DESCRIPTION:
                     01D8   553 ;       Convert D_FLOATING to H_FLOATING.
                     01D8   554 ;
                     01D8   555 ;
                     01D8   556 ; CALLING SEQUENCE:
                     01D8   557 ;       LIB$$CVT_CVTDH_R1 ( dfloat.rd.r, hfloat.wh.r )
                     01D8   558 ;       This is a JSB entry point.
                     01D8   559 ;
                     01D8   560 ;
                     01D8   561 ; FORMAL PARAMETERS:
                     01D8   562 ;       R0 --> dfloat   R1 --> hfloat
                     01D8   563 ;
                     01D8   564 ;
                     01D8   565 ; IMPLICIT INPUTS:
                     01D8   566 ;       NONE
                     01D8   567 ;
                     01D8   568 ;
                     01D8   569 ; IMPLICIT OUTPUTS:
                     01D8   570 ;       NONE
                     01D8   571 ;
                     01D8   572 ;
                     01D8   573 ; COMPLETION CODES:
                     01D8   574 ;       NONE
                     01D8   575 ;
                     01D8   576 ; SIDE EFFECTS:
                     01D8   577 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     01D8   578 ;
                     01D8   579 ;--
                     01D8   580
                     01D8   581 LIB$$CVT_CVTDH_R1 ::
 61    60 32FD       01D8   582          CVTDH   (R0), (R1)              ;Convert D_FLOATING to H_FLOATING
       05            01DC   583          RSB
```

```
                     01DD   585          .SBTTL  LIB$$CVT_CVTRHL_R1
                     01DD   586
                     01DD   587 ;++
                     01DD   588 ; FUNCTIONAL DESCRIPTION:
                     01DD   589 ;       Convert H_FLOATING to LONGWORD rounded.
                     01DD   590 ;
                     01DD   591 ;
                     01DD   592 ; CALLING SEQUENCE:
                     01DD   593 ;       LIB$$CVT_CVTRHL_R1 ( hfloat.rh.r, long.wl.r )
                     01DD   594 ;       This is a JSB entry point.
                     01DD   595 ;
                     01DD   596 ;
                     01DD   597 ; FORMAL PARAMETERS:
                     01DD   598 ;       R0 --> hfloat   R1 --> long
                     01DD   599 ;
                     01DD   600 ;
                     01DD   601 ; IMPLICIT INPUTS:
                     01DD   602 ;       NONE
                     01DD   603 ;
                     01DD   604 ;
                     01DD   605 ; IMPLICIT OUTPUTS:
                     01DD   606 ;       NONE
                     01DD   607 ;
                     01DD   608 ;
                     01DD   609 ; COMPLETION CODES:
                     01DD   610 ;       NONE
                     01DD   611 ;
                     01DD   612 ; SIDE EFFECTS:
                     01DD   613 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     01DD   614 ;
                     01DD   615 ;--
                     01DD   616 LIB$$CVT_CVTRHL_R1 ::
61      60 6BFD      01DD   617          CVTRHL  (R0), (R1)          ;Convert H_FLOATING to LONGWORD rounded
           05        01E1   618          RSB
```

```
                     01E2    620          .SBTTL  LIB$$CVT_CVTRHQ_R1
                     01E2    621 ;++
                     01E2    622 ; FUNCTIONAL DESCRIPTION:
                     01E2    623 ;       Convert H_FLOATING to QUADWORD rounded.
                     01E2    624 ;
                     01E2    625 ;
                     01E2    626 ; CALLING SEQUENCE:
                     01E2    627 ;       LIB$$CVT_CVTRHQ_R1 ( hfloat.rh.r, quad.wq.r )
                     01E2    628 ;       This is a JSB entry point.
                     01E2    629 ;
                     01E2    630 ;
                     01E2    631 ; FORMAL PARAMETERS:
                     01E2    632 ;       R0 --> hfloat   R1 --> quad
                     01E2    633 ;
                     01E2    634 ;
                     01E2    635 ; IMPLICIT INPUTS:
                     01E2    636 ;       NONE
                     01E2    637 ;
                     01E2    638 ;
                     01E2    639 ; IMPLICIT OUTPUTS:
                     01E2    640 ;       NONE
                     01E2    641 ;
                     01E2    642 ;
                     01E2    643 ; COMPLETION CODES:
                     01E2    644 ;       NONE
                     01E2    645 ;
                     01E2    646 ; SIDE EFFECTS:
                     01E2    647 ;
                     01E2    648 ;
                     01E2    649 ;--
                     01E2    650 LIB$$CVT_CVTRHQ_R1 ::
                 0F  BB  01E2    651          PUSHR   #^M<R0,R1,R2,R3>        ; Save these registers
                 61  7C  01E4    652          CLRQ    (R1)                    ; Initialize result to zero
                 53  D4  01E6    653          CLRL    R3                      ; Initialize rounding bit
                     01E8    654 ;
                     01E8    655 ;        Find the exponent
                     01E8    656 ;
          52 60  0F  00  EF  01E8    657          EXTZV   #0, #15, (R0), R2       ; Extract the exponent
                     0D  13  01ED    658          BEQL    5$                      ; Call the number zero if expo. is zero
           52  00004000  8F  C2  01EF    659          SUBL2   #^X4000, R2             ; Take out the excess
                     04  19  01F6    660          BLSS    5$                      ; If negative then Quad is zero
                     05  14  01F8    661          BGTR    10$                     ; Is exponent zero ?
                     61  D6  01FA    662          INCL    (R1)                    ; Yes, call it a one
                     01FC    663 5$:
                 007D  31  01FC    664          BRW     50$                     ; Go to the finish line
                     01FF    665 ;
                     01FF    666 ;        Check for overflow.  If exponent is 63, find rounding bit before we
                     01FF    667 ;        lose it.
                     01FF    668 ;
                     01FF    669 10$:
                 52  3F  C2  01FF    670          SUBL2   #63, R2                 ; Bring '.' to right of bit 0
                     1E  19  0202    671          BLSS    20$                     ; If exponent < 63, skip this
                     0F  14  0204    672          BGTR    15$                     ; If exponent > 63, we have IV
    53 08 A0  01  01  EF  0206    673          EXTZV   #1, #1, 8(R0), R3       ; Exponent is 63, R3 is the round bit
           53  80000000  8F  CA  020C    674          BICL2   #^X80000000, R3         ; Turn on the 31st bit of R3 to flag
                     0213    675                                                  ; that the round bit has been determined
                     0D  11  0213    676          BRB     20$                     ; Cont.
```

```
                                0215   677 15$:
          0000047C 8F   DD      0215   678          PUSHL   #SS$_INTOVF                ; Signal IV
00000000'GF    01   FB      021B   679          CALLS   #1, G^LIB$SIGNAL          ;
                                0222   680 ;
                                0222   681 ;       Move the fraction to the QUAD, and put in a positive sign and MSB
                                0222   682 ;
                                0222   683 20$:
       61    08 A0   B0      0222   684          MOVW    8(R0), (R1)               ; Move the fraction to QUAD
    02 A1    06 A0   B0      0226   685          MOVW    6(R0), 2(R1)              ;
    04 A1    04 A0   B0      022B   686          MOVW    4(R0), 4(R1)              ;
    06 A1    02 A0   B0      0230   687          MOVW    2(R0), 6(R1)              ;
 61    61    FE 8F   79      0235   688          ASHQ    #-2, (R1), (R1)           ; Make room for sign and MSB
    07 A1    C0 8F   8A      023A   689          BICB2   #^XC0, 7(R1)              ; Make sure two high bits are off
    07 A1    40 8F   88      023F   690          BISB2   #^X40, 7(R1)              ; Sign is '+', MSB is 1
                                0244   691 ;
                                0244   692 ;       Find rounding bit if not already done
                                0244   693 ;
                      53    D5      0244   694          TSTL    R3                        ; Is rounding bit already found ?
                      0A    19      0246   695          BLSS    30$                       ; Yes
                53    52    CE      0248   696          MNEGL   R2, R3                    ; R3 will contain rounding bit
                      53    D7      024B   697          DECL    R3                        ; Look at the right of decimal point
 53    61    01    53    EF      024D   698          EXTZV   R3, #1, (R1), R3          ; This is the rounding bit
                                0252   699 ;
                                0252   700 ;       Shift the QAUD R2 times
                                0252   701 ;
                                0252   702 30$:
       61    61    52    79      0252   703          ASHQ    R2, (R1), (R1)            ; Shift the QUAD R2 times
                      53    95      0256   704          TSTB    R3                        ; Do we need to round
                      0B    13      0258   705          BEQL    40$                       ; No
          52    61    D0      025A   706          MOVL    (R1), R2                  ; Do the rounding
    61    01    A2    9E      025D   707          MOVAB   1(R2), (R1)               ; ...
    04 A1    00    D8      0261   708          ADWC    #0, 4(R1)                 ; ...
                      60    B5      0265   709 40$:    TSTW    (R0)                      ; Is the H negative ?
                      13    14      0267   710          BGTR    50$                       ; No
       61    61    D2      0269   711          MCOML   (R1), (R1)                ; Negate the QUAD
 04 A1    04 A1    D2      026C   712          MCOML   4(R1), 4(R1)               ; ...
          50    61    D0      0271   713          MOVL    (R1), R0                  ; ...
    61    01    A0    9E      0274   714          MOVAB   1(R0), (R1)               ; ...
    04 A1    00    D8      0278   715          ADWC    #0, 4(R1)                 ; ...
                                027C   716 50$:
                      0F    BA      027C   717          POPR    #^M<R0,R1,R2,R3>          ; Restore these registers
                            05      027E   718          RSB
```

```
                     027F    720          .SBTTL  LIB$$CVT_CVTRHO_R1
                     027F    721 ;++
                     027F    722 ; FUNCTIONAL DESCRIPTION:
                     027F    723 ;       Convert H_FLOATING to OCTAWORD rounded.
                     027F    724 ;
                     027F    725 ;
                     027F    726 ; CALLING SEQUENCE:
                     027F    727 ;       LIB$$CVT_CVTRHO_R1 ( hfloat.rh.r, octa.wo.r )
                     027F    728 ;       This is a JSB entry point.
                     027F    729 ;
                     027F    730 ;
                     027F    731 ; FORMAL PARAMETERS:
                     027F    732 ;       R0 --> hfloat   R1 --> octa
                     027F    733 ;
                     027F    734 ;
                     027F    735 ; IMPLICIT INPUTS:
                     027F    736 ;       NONE
                     027F    737 ;
                     027F    738 ;
                     027F    739 ; IMPLICIT OUTPUTS:
                     027F    740 ;       NONE
                     027F    741 ;
                     027F    742 ;
                     027F    743 ; COMPLETION CODES:
                     027F    744 ;       NONE
                     027F    745 ;
                     027F    746 ; SIDE EFFECTS:
                     027F    747 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     027F    748 ;
                     027F    749 ;--
00000004             027F    750 O_LEN   = 4                 ; Length in longwords (octaword)
                     027F    751
                     027F    752 LIB$$CVT_CVTRHO_R1 ::
                     027F    753         ;
                     027F    754         ; Initialization
                     027F    755         ;
                  3F   BB    027F    756         PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; Save these registers
                  61   7C    0281    757         CLRQ    (R1)                    ; Initialize result to zero
               08 A1   7C    0283    758         CLRQ    8(R1)                   ; Initialize result to zero
            7E    60 70FD    0286    759         MOVH    (R0), -(SP)             ; Grab source (reserved operand?)
                  43   13    028A    760         BEQL    30$                     ; Branch if result is zero
                     028C    761         ;
                     028C    762         ; The source is not zero; initialize other registers
                     028C    763         ;
                  51   10   C0    028C    764         ADDL2   #O_LEN*4, R1            ; Address past octaword
                  54   51   D0    028F    765         MOVL    R1, R4                  ; Save this address
                  55   03   D0    0292    766         MOVL    #O_LEN-1, R5            ; Loop counter
                     0295    767         ;
                     0295    768 10$:    ; Loop for each longword of destination
                     0295    769         ;
52                55   05   78    0295    770         ASHL    #5, R5, R2              ; Scaling amount (2**5 = 32)
                  53   51   D0    0299    771         MOVL    R1, R3                  ; Address where we start decrementing
                  6E   52   A2    029C    772         SUBW2   R2, (SP)                ; Scale down by 2**(32*n)
                  71   6E 6BFD    029F    773         CVTRHL  (SP), -(R1)             ; Store high-order longword of result
                  6E   52   A0    02A3    774         ADDW2   R2, (SP)                ; Scale up by 2**(32*n)
                  7E   61 6EFD    02A6    775         CVTLH   (R1), -(SP)             ; Convert long back to double
                  1C   13    02AA    776         BEQL    20$                     ; Skip if zero
```

```
                     10   11  02AC    777          BRB     13$                     ; Jump into decrement loop
                                  02AE    778          ;
                                  02AE    779          ; If the longword was negative, we must sign-extend the result
                                  02AE    780          ;
            80000000 8F   63   D1  02AE    781 11$:     CMPL    (R3), #^X80000000       ; Compare with largest negative number
                          05   12  02B5    782          BNEQ    12$                     ; If not equal, do a simple decrement
                     83   63   D2  02B7    783          MCOML   (R3), (R3)+             ; Subtract one and consider negative
                          04   11  02BA    784          BRB     14$                     ; Stay in loop, as result is negative
                          83   D7  02BC    785 12$:     DECL    (R3)+                   ; Decrement this longword
                          05   18  02BE    786 13$:     BGEQ    15$                     ; Jump out of loop if positive
                     54   53   D1  02C0    787 14$:     CMPL    R3, R4                  ; See if we are done decrementing
                          E9   1F  02C3    788          BLSSU   11$                     ; If not past octaword, keep going
                                  02C5    789          ;
                                  02C5    790          ; Scale up the longword (if not zero), and subtract from stack temp
                                  02C5    791          ;
                     6E   52   A0  02C5    792 15$:     ADDW2   R2, (SP)                ; Scale up by 2**(32*n)
                     6E   8E 62FD  02C8    793 20$:     SUBH2   (SP)+, (SP)             ; Subtract high-order longword
                                  02CC    794          ;
                                  02CC    795          ; Decrease loop counter, and continue, if more longwords required
                                  02CC    796          ;
                       C6 55   F4  02CC    797          SOBGEQ  R5, 10$                 ; Continue looping
                                  02CF    798          ;
                                  02CF    799 30$:     ; Clean up the stack, and restore registers
                                  02CF    800          ;
                          8E 7CFD 02CF    801          CLRH    (SP)+                   ; Pop huge from the stack
                          3F   BA 02D2    802          POPR    #^M<R0,R1,R2,R3,R4,R5>  ; Restore these registers
                               05 02D4    803          RSB                             ; Return from subroutine
                                  02D5    804
```

```
                     02D5   806          .SBTTL  LIB$$CVT_CVTHF_R1
                     02D5   807
                     02D5   808 ;++
                     02D5   809 ; FUNCTIONAL DESCRIPTION:
                     02D5   810 ;       Convert H_FLOATING to FLOATING
                     02D5   811 ;
                     02D5   812 ;
                     02D5   813 ; CALLING SEQUENCE:
                     02D5   814 ;       LIB$$CVT_CVTHF_R1 (hfloat.rh.r, float.wf.r )
                     02D5   815 ;       This is a JSB entry point.
                     02D5   816 ;
                     02D5   817 ;
                     02D5   818 ; FORMAL PARAMETERS:
                     02D5   819 ;       R0 --> hfloat   R1 --> float
                     02D5   820 ;
                     02D5   821 ;
                     02D5   822 ; IMPLICIT INPUTS:
                     02D5   823 ;       NONE
                     02D5   824 ;
                     02D5   825 ;
                     02D5   826 ; IMPLICIT OUTPUTS:
                     02D5   827 ;       NONE
                     02D5   828 ;
                     02D5   829 ;
                     02D5   830 ; COMPLETION CODES:
                     02D5   831 ;       NONE
                     02D5   832 ;
                     02D5   833 ; SIDE EFFECTS:
                     02D5   834 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     02D5   835 ;
                     02D5   836 ;--
                     02D5   837 LIB$$CVT_CVTHF_R1 ::
61       60 F6FD     02D5   838          CVTHF   (R0), (R1)              ;Convert H_FLOATING to FLOAT
                05   02D9   839          RSB
```

```
                     02DA   841          .SBTTL  LIB$$CVT_CVTHD_R1
                     02DA   842
                     02DA   843 ;++
                     02DA   844 ; FUNCTIONAL DESCRIPTION:
                     02DA   845 ;       Convert H_FLOATING to D_FLOATING
                     02DA   846 ;
                     02DA   847 ;
                     02DA   848 ; CALLING SEQUENCE:
                     02DA   849 ;       LIB$$CVT_CVTHD_R1 ( hfloat.rh.r, dfloat.wd.r )
                     02DA   850 ;       This is a JSB entry point.
                     02DA   851 ;
                     02DA   852 ;
                     02DA   853 ; FORMAL PARAMETERS:
                     02DA   854 ;       R0 --> hfloat   R1 --> dfloat
                     02DA   855 ;
                     02DA   856 ;
                     02DA   857 ; IMPLICIT INPUTS:
                     02DA   858 ;       NONE
                     02DA   859 ;
                     02DA   860 ;
                     02DA   861 ; IMPLICIT OUTPUTS:
                     02DA   862 ;       NONE
                     02DA   863 ;
                     02DA   864 ;
                     02DA   865 ; COMPLETION CODES:
                     02DA   866 ;       NONE
                     02DA   867 ;
                     02DA   868 ; SIDE EFFECTS:
                     02DA   869 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     02DA   870 ;
                     02DA   871 ;--
                     02DA   872 LIB$$CVT_CVTHD_R1 ::
61   60 F7FD         02DA   873          CVTHD   (R0), (R1)          ;Convert H_FLOATING to D_FLOATING
         05          02DE   874          RSB
```

```
                                .SBTTL  LIB$$CVT_CVTHG_R1

                        ;++
                        ; FUNCTIONAL DESCRIPTION:
                        ;       Convert H_FLOATING to G_FLOATING
                        ;
                        ; CALLING SEQUENCE:
                        ;       LIB$$CVT_CVTHG_R1 ( hfloat.rh.r, gfloat.wg.r )
                        ;       This is a JSB entry point.
                        ;
                        ; FORMAL PARAMETERS:
                        ;       R0 --> hfloat   R1 --> gfloat
                        ;
                        ; IMPLICIT INPUTS:
                        ;       NONE
                        ;
                        ; IMPLICIT OUTPUTS:
                        ;       NONE
                        ;
                        ; COMPLETION CODES:
                        ;       NONE
                        ;
                        ; SIDE EFFECTS:
                        ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                        ;
                        ;--
                        LIB$$CVT_CVTHG_R1 ::
61    60 76FD           CVTHG   (R0), (R1)              ;Convert H_FLOATING to G_FLOATING
         05             RSB
```

```
                     02E4    911          .SBTTL  LIBS$CVT_CVTGH_R1
                     02E4    912
                     02E4    913 ;++
                     02E4    914 ; FUNCTIONAL DESCRIPTION:
                     02E4    915 ;       Convert G_FLOATING to H_FLOATING
                     02E4    916 ;
                     02E4    917 ;
                     02E4    918 ; CALLING SEQUENCE:
                     02E4    919 ;       LIBS$CVT_CVTGH_R1 ( gfloat.rg.r, hfloat.wh.r )
                     02E4    920 ;       This is a JSB entry point.
                     02E4    921 ;
                     02E4    922 ;
                     02E4    923 ; FORMAL PARAMETERS:
                     02E4    924 ;       R0 --> gfloat   R1 --> hfloat
                     02E4    925 ;
                     02E4    926 ;
                     02E4    927 ; IMPLICIT INPUTS:
                     02E4    928 ;       NONE
                     02E4    929 ;
                     02E4    930 ;
                     02E4    931 ; IMPLICIT OUTPUTS:
                     02E4    932 ;       NONE
                     02E4    933 ;
                     02E4    934 ;
                     02E4    935 ; COMPLETION CODES:
                     02E4    936 ;       NONE
                     02E4    937 ;
                     02E4    938 ; SIDE EFFECTS:
                     02E4    939 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     02E4    940 ;
                     02E4    941 ;--
                     02E4    942 LIBS$CVT_CVTGH_R1 ::
  61    60 56FD      02E4    943          CVTGH   (R0), (R1)              ;Convert G_FLOATING TO H_FLOATING
        05           02E8    944          RSB
```

```
                                        .SBTTL  LIB$$CVT_SCALE_OU_UP_BY_10_R1

                                ;++
                                ; FUNCTIONAL DESCRIPTION:
                                ;       Scale an unsigned OCTAWORD up by 10.
                                ;
                                ;
                                ; CALLING SEQUENCE:
                                ;       LIB$$CVT_SCALE_OU_UP_BY_10_R1 ( octa.mo.r )
                                ;       This is a JSB entry point.
                                ;
                                ;
                                ; FORMAL PARAMETERS:
                                ;       R0 --> octa
                                ;
                                ;
                                ; IMPLICIT INPUTS:
                                ;       NONE
                                ;
                                ;
                                ; IMPLICIT OUTPUTS:
                                ;       NONE
                                ;
                                ;
                                ; COMPLETION CODES:
                                ;       NONE
                                ;
                                ; SIDE EFFECTS:
                                ;
                                ;       NONE
                                ;--
                                LIB$$CVT_SCALE_OU_UP_BY_10_R1 ::
                        1F  BB          PUSHR   #^M<R0,R1,R2,R3,R4>
                    54  04  D0          MOVL    #4, R4                  ; Do four longwords
                        53  D4          CLRL    R3                      ; Clear the 'carry'
                51  53  60  0A  7A      EMUL    #10, (R0), R3, R1       ; Multiply low-order longwords
                        03  18          BGEQ    20$                     ; Make sure low-order longword
                    52  0A  C0          ADDL2   #10, R2                 ;    is seen as unsigned
                    80  51  D0  02FA    MOVL    R1, (R0)+               ; Store low longword
                    53  52  D0  02FD    MOVL    R2, R3                  ; Save the 'carry'
                      ED  54  F5  0300  SOBGTR  R4, 10$                 ; Continue looping
                        53  D5  0303    TSTL    R3                      ; Check for overflow
                        0E  13  0305    BEQL    30$                     ; Branch if no overflow
             7E     0000047C  8F  D0  0307  MOVL    #SS$_INTOVF, -(SP)      ; Integer overflow
             00000000'GF      01  FB  030E  CALLS   #1, G^LIB$SIGNAL        ; Signal the error
                        1F  BA  0315    POPR    #^M<R0,R1,R2,R3,R4>
                            05  0317    RSB                             ; Return
                                0318
```

```
                                                0318   995          .SBTTL  LIB$$CVT_SCALE_OU_DOWN_BY_10_R1
                                                0318   996
                                                0318   997 ;++
                                                0318   998 ; FUNCTIONAL DESCRIPTION:
                                                0318   999 ;       Scales down an OCTAWORD by 10.
                                                0318  1000 ;
                                                0318  1001 ;
                                                0318  1002 ; CALLING SEQUENCE:
                                                0318  1003 ;       LIB$$CVT_SCALE_OU_DOWN_BY_10_R1 ( octa.mo.r )
                                                0318  1004 ;       This is a JSB entry point.
                                                0318  1005 ;
                                                0318  1006 ;
                                                0318  1007 ; FORMAL PARAMETERS:
                                                0318  1008 ;       R0 --> octa
                                                0318  1009 ;
                                                0318  1010 ;
                                                0318  1011 ; IMPLICIT INPUTS:
                                                0318  1012 ;       NONE
                                                0318  1013 ;
                                                0318  1014 ;
                                                0318  1015 ; IMPLICIT OUTPUTS:
                                                0318  1016 ;       NONE
                                                0318  1017 ;
                                                0318  1018 ;
                                                0318  1019 ; COMPLETION CODES:
                                                0318  1020 ;       NONE
                                                0318  1021 ;
                                                0318  1022 ; SIDE EFFECTS:
                                                0318  1023 ;       NONE
                                                0318  1024 ;
                                                0318  1025 ;--
                                                0318  1026 LIB$$CVT_SCALE_OU_DOWN_BY_10_R1::
                                     1F   BB    0318  1027          PUSHR   #^M<R0,R1,R2,R3,R4>
                                   54 03   D0   031A  1028          MOVL    #3, R4                  ; Do four longwords
                                      52   D4   031D  1029          CLRL    R2                      ; Clear high longword of quad
                                   51 6044   D0   031F  1030 10$:    MOVL    (R0)[R4], R1            ; Grab low longword of quad
52   6044                       51 0A   7B   0323  1031          EDIV    #10, R1, (R0)[R4], R2   ; Do a divide
                                   52 05   D1   0329  1032 30$:    CMPL    #5, R2                  ; See if remainder too large
                                      0D   1A   032C  1033          BGTRU   50$                     ;    for the next EDIV
                                      08   15   032E  1034          BLEQ    40$                     ; Simply make remainder smaller
                                   52 0A   C0   0330  1035          ADDL2   #10, R2                 ; Make remainder positive
                                 6044      D7   0333  1036          DECL    (R0)[R4]                ; Make quotient smaller
                                      F1   11   0336  1037          BRB     30$                     ; Join the looping code
                                   52 0A   C2   0338  1038 40$:    SUBL2   #10, R2                 ; Make remainder smaller
                                   E1 54   F4   033B  1039 50$:    SOBGEQ  R4, 10$                 ; Continue looping
                                      1F   BA   033E  1040          POPR    #^M<R0,R1,R2,R3,R4>
                                           05   0340  1041          RSB                             ; Return
                                                0341  1042
```

```
                     0341  1044          .SBTTL  LIB$$CVT_MULD2_R1
                     0341  1045
                     0341  1046 ;++
                     0341  1047 ; FUNCTIONAL DESCRIPTION:
                     0341  1048 ;       multiply two DOUBLE floating values.
                     0341  1049 ;
                     0341  1050 ;
                     0341  1051 ; CALLING SEQUENCE:
                     0341  1052 ;       LIB$$CVT_MULD2_R1 ( mulr.rd.r, prod.md.r )
                     0341  1053 ;       This is a JSB entry point.
                     0341  1054 ;
                     0341  1055 ;
                     0341  1056 ; FORMAL PARAMETERS:
                     0341  1057 ;       R0 --> mulr     R1 --> prod
                     0341  1058 ;
                     0341  1059 ;
                     0341  1060 ; IMPLICIT INPUTS:
                     0341  1061 ;       NONE
                     0341  1062 ;
                     0341  1063 ;
                     0341  1064 ; IMPLICIT OUTPUTS:
                     0341  1065 ;       NONE
                     0341  1066 ;
                     0341  1067 ;
                     0341  1068 ; COMPLETION CODES:
                     0341  1069 ;       NONE
                     0341  1070 ;
                     0341  1071 ; SIDE EFFECTS:
                     0341  1072 ;       NONE
                     0341  1073 ;
                     0341  1074 ;--
                     0341  1075 LIB$$CVT_MULD2_R1 ::
 61    60    64      0341  1076          MULD2   (R0), (R1)              ;Multiply two D_FLOATING
                05   0344  1077          RSB
```

```
                     0345  1079          .SBTTL  LIB$$CVT_DIVD2_R1
                     0345  1080
                     0345  1081 ;++
                     0345  1082 ; FUNCTIONAL DESCRIPTION:
                     0345  1083 ;       Divide two DOUBLE floating values.
                     0345  1084 ;
                     0345  1085 ;
                     0345  1086 ; CALLING SEQUENCE:
                     0345  1087 ;       LIB$$CVT_DIVD2_R1 ( divr.rd.r, quo.md.r )
                     0345  1088 ;       This is a JSB entry point.
                     0345  1089 ;
                     0345  1090 ;
                     0345  1091 ; FORMAL PARAMETERS:
                     0345  1092 ;       R0 --> divr     R1 --> quo
                     0345  1093 ;
                     0345  1094 ;
                     0345  1095 ; IMPLICIT INPUTS:
                     0345  1096 ;       NONE
                     0345  1097 ;
                     0345  1098 ;
                     0345  1099 ; IMPLICIT OUTPUTS:
                     0345  1100 ;       NONE
                     0345  1101 ;
                     0345  1102 ;
                     0345  1103 ; COMPLETION CODES:
                     0345  1104 ;       NONE
                     0345  1105 ;
                     0345  1106 ; SIDE EFFECTS:
                     0345  1107 ;       NONE
                     0345  1108 ;
                     0345  1109 ;--
                     0345  1110 LIB$$CVT_DIVD2_R1 ::
  61    60    66     0345  1111          DIVD2   (R0), (R1)              ;Divide two D_FLOATING
                05   0348  1112          RSB
```

```
                     0349  1114          .SBTTL  LIB$$CVT_MULH2_R1
                     0349  1115
                     0349  1116 ;++
                     0349  1117 ; FUNCTIONAL DESCRIPTION:
                     0349  1118 ;       multiply two H_FLOATING floating values.
                     0349  1119 ;
                     0349  1120 ;
                     0349  1121 ; CALLING SEQUENCE:
                     0349  1122 ;       LIB$$CVT_MULH2_R1 ( mulr.rh.r, prod.mh.r )
                     0349  1123 ;       This is a JSB entry point.
                     0349  1124 ;
                     0349  1125 ;
                     0349  1126 ; FORMAL PARAMETERS:
                     0349  1127 ;       R0 --> mulr     R1 --> prod
                     0349  1128 ;
                     0349  1129 ;
                     0349  1130 ; IMPLICIT INPUTS:
                     0349  1131 ;       NONE
                     0349  1132 ;
                     0349  1133 ;
                     0349  1134 ; IMPLICIT OUTPUTS:
                     0349  1135 ;       NONE
                     0349  1136 ;
                     0349  1137 ;
                     0349  1138 ; COMPLETION CODES:
                     0349  1139 ;       NONE
                     0349  1140 ;
                     0349  1141 ; SIDE EFFECTS:
                     0349  1142 ;
                     0349  1143 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     0349  1144 ;
                     0349  1145 ;--
                     0349  1146 LIB$$CVT_MULH2_R1 ::
61     60 64FD       0349  1147          MULH2   (R0), (R1)              ;Multiply two H_FLOATING
         05          034D  1148          RSB
```

```
                     034E  1150          .SBTTL  LIB$$CVT_DIVH2_R1
                     034E  1151
                     034E  1152 ;++
                     034E  1153 ; FUNCTIONAL DESCRIPTION:
                     034E  1154 ;       Divide two H_FLOATING floating values.
                     034E  1155 ;
                     034E  1156 ;
                     034E  1157 ; CALLING SEQUENCE:
                     034E  1158 ;       LIB$$CVT_DIVH2_R1 ( divr.rh.r, quo.mh.r )
                     034E  1159 ;       This is a JSB entry point.
                     034E  1160 ;
                     034E  1161 ;
                     034E  1162 ; FORMAL PARAMETERS:
                     034E  1163 ;       R0 --> divr     R1 --> quo
                     034E  1164 ;
                     034E  1165 ;
                     034E  1166 ; IMPLICIT INPUTS:
                     034E  1167 ;       NONE
                     034E  1168 ;
                     034E  1169 ;
                     034E  1170 ; IMPLICIT OUTPUTS:
                     034E  1171 ;       NONE
                     034E  1172 ;
                     034E  1173 ;
                     034E  1174 ; COMPLETION CODES:
                     034E  1175 ;       NONE
                     034E  1176 ;
                     034E  1177 ; SIDE EFFECTS:
                     034E  1178 ;
                     034E  1179 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                     034E  1180 ;
                     034E  1181 ;--
                     034E  1182 LIB$$CVT_DIVH2_R1 ::
61    60 66FD        034E  1183          DIVH2   (R0), (R1)              ;Divide two H_FLOATING
          05         0352  1184          RSB
```

```
                          0353  1186          .SBTTL  LIB$$CVT_ASHP_R1
                          0353  1187
                          0353  1188 ;++
                          0353  1189 ; FUNCTIONAL DESCRIPTION:
                          0353  1190 ;       Scale a packed decimal number.
                          0353  1191 ;
                          0353  1192 ;
                          0353  1193 ; CALLING SEQUENCE:
                          0353  1194 ;       LIB$$CVT_ASHP_R1 (cnt.rw.r, srclen.rw.r, srcaddr.ra.r,
                          0353  1195 ;                         round.rb.r, dstlen.rw.r, dstaddr.ra.r )
                          0353  1196 ;       This is a JSB entry point.
                          0353  1197 ;
                          0353  1198 ;
                          0353  1199 ; FORMAL PARAMETERS:
                          0353  1200 ;       R0 --> cnt      R1 = srclen     R2 --> srcaddr  R3 = round
                          0353  1201 ;       R4 --> dstlen   R5 = dstaddr
                          0353  1202 ;
                          0353  1203 ;
                          0353  1204 ; IMPLICIT INPUTS:
                          0353  1205 ;       NONE
                          0353  1206 ;
                          0353  1207 ;
                          0353  1208 ; IMPLICIT OUTPUTS:
                          0353  1209 ;       NONE
                          0353  1210 ;
                          0353  1211 ;
                          0353  1212 ; COMPLETION CODES:
                          0353  1213 ;       NONE
                          0353  1214 ;
                          0353  1215 ; SIDE EFFECTS:
                          0353  1216 ;       NONE
                          0353  1217 ;
                          0353  1218 ;--
                          0353  1219 LIB$$CVT_ASHP_R1 ::
                    3F    BB    0353  1220          PUSHR   #^M<R0,R1,R2,R3,R4,R5>
65    64    63    62    61  60    F8    0355  1221          ASHP    (R0), (R1), (R2), (R3), (R4), (R5)
                    3F    BA    035C  1222          POPR    #^M<R0,R1,R2,R3,R4,R5>
                          05    035E  1223          RSB
```

```
                                035F  1225          .SBTTL  LIB$$CVT_CMPH_R1
                                035F  1226
                                035F  1227 ;++
                                035F  1228 ; FUNCTIONAL DESCRIPTION:
                                035F  1229 ;       Compare two H floating values
                                035F  1230 ;
                                035F  1231 ;
                                035F  1232 ; CALLING SEQUENCE:
                                035F  1233 ;       STATUS = LIB$$CVT_CMPH_R1 ( src1.rh.r, src2.rh.r )
                                035F  1234 ;       This is a JSB entry point.
                                035F  1235 ;
                                035F  1236 ;
                                035F  1237 ; FORMAL PARAMETERS:
                                035F  1238 ;       R0 --> src1     R1 --> src2
                                035F  1239 ;
                                035F  1240 ;
                                035F  1241 ; IMPLICIT INPUTS:
                                035F  1242 ;       NONE
                                035F  1243 ;
                                035F  1244 ;
                                035F  1245 ; IMPLICIT OUTPUTS:
                                035F  1246 ;       NONE
                                035F  1247 ;
                                035F  1248 ;
                                035F  1249 ; COMPLETION CODES:
                                035F  1250 ;       R0 = -1 if src1 < src2
                                035F  1251 ;       R0 = 0  if src1 = src2
                                035F  1252 ;       R0 = 1  if src1 > src2
                                035F  1253 ;
                                035F  1254 ; SIDE EFFECTS:
                                035F  1255 ;       OPCODE RESERVED TO DIGITAL error is possible, see ENVIRONMENT.
                                035F  1256 ;
                                035F  1257 ;--
                                035F  1258 LIB$$CVT_CMPH_R1 ::
          61    60 71FD         035F  1259          CMPH    (R0), (R1)
                06   13         0363  1260          BEQL    10$
                07   19         0365  1261          BLSS    20$
          50    01   CE         0367  1262          MNEGL   #1, R0                  ;src1 > src2
                     05         036A  1263          RSB                             ;Finished
                                036B  1264 10$:
                50   D4         036B  1265          CLRL    R0                      ;They are equal
                     05         036D  1266          RSB                             ;Finished
                                036E  1267 20$:
50    FFFFFFFF 8F   D0         036E  1268          MOVL    #-1, R0                 ;src1 < src2
                     05         0375  1269          RSB                             ;Finished
                                0376  1270
```

```
0376  1272 .END
```

```
D32                             00000062 R      01
DDUMMY                          0000004A R      01
D_OFF                         = 00000007
F32                             00000019 R      01
FDUMMY                          0000000D R      01
G32                             000000B3 R      01
GDUMMY                          0000009B R      01
LIB$$CVT_ASHP_R1                00000353 RG     01
LIB$$CVT_CMPH_R1                0000035F RG     01
LIB$$CVT_CVTDH_R1               000001D8 RG     01
LIB$$CVT_CVTGH_R1               000002E4 RG     01
LIB$$CVT_CVTHD_R1               000002DA RG     01
LIB$$CVT_CVTHF_R1               000002D5 RG     01
LIB$$CVT_CVTHG_R1               000002DF RG     01
LIB$$CVT_CVTLB_R1               00000000 RG     01
LIB$$CVT_CVTLH_R1               00000008 RG     01
LIB$$CVT_CVTLW_R1               00000004 RG     01
LIB$$CVT_CVTRDB_R1              000001A8 RG     01
LIB$$CVT_CVTRHL_R1              000001DD RG     01
LIB$$CVT_CVTRHO_R1              0000027F RG     01
LIB$$CVT_CVTRHQ_R1              000001E2 RG     01
LIB$$CVT_CVTROUD_R1             00000072 RG     01
LIB$$CVT_CVTROUF_R1             00000021 RG     01
LIB$$CVT_CVTROUG_R1             000000C3 RG     01
LIB$$CVT_CVTROUH_R1             000000F0 RG     01
LIB$$CVT_DIVD2_R1               00000345 RG     01
LIB$$CVT_DIVH2_R1               0000034E RG     01
LIB$$CVT_MULD2_R1               00000341 RG     01
LIB$$CVT_MULH2_R1               00000349 RG     01
LIB$$CVT_SCALE_OU_DOWN_BY_10_R1 00000318 RG     01
LIB$$CVT_SCALE_OU_UP_BY_10_R1   000002E9 RG     01
LIB$SIGNAL                      ******** X      01
O_LEN                         = 00000004
SHIFT_OU_LEFT                   0000018C R      01
SS$_INTOVF                    = 0000047C
```

Psect synopsis

PSECT name	Allocation		PSECT No.	Attributes
. ABS .	00000000	(0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
LIB$CODE	00000376	(886.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG
ABS	00000000	(0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.04	00:00:02.60
Command processing	121	00:00:00.33	00:00:05.82
Pass 1	229	00:00:03.60	00:00:25.63
Symbol table sort	0	00:00:00.47	00:00:05.50
Pass 2	209	00:00:01.68	00:00:07.80

Symbol table output	5	00:00:00.04	00:00:00.04
Psect synopsis output	3	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	598	00:00:06.18	00:00:47.40

The working set limit was 1200 pages.
34798 bytes (68 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 441 non-local and 44 local symbols.
1272 source lines were read in Pass 1, producing 12 object records in Pass 2.
8 pages of virtual memory were used to define 7 macros.

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

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	4

469 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

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

LIBCVTDX
LIS

LIBCVTMAC
LIS

LIBCRCTAB
LIS

LIBCVTDF
LIS

LIBCUSTOM
LIS

LIBCREDIR
LIS

LIBCUTATB
LIS

LIBCOMMON
LIS

LIBCRC
LIS

0205 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY