COBRTL

```
CCCCCCCC    000000   BBBBBBBB    CCCCCCC  VV      VV  TTTTTTTTTT   QQQQQQ    FFFFFFFFFF
CCCCCCCC    000000   BBBBBBBB    CCCCCCC  VV      VV  TTTTTTTTTT   QQQQQQ    FFFFFFFFFF
CC         00    00  BB      BB  CC        VV    VV       TT      QQ    QQ   FF
CC         00    00  BB      BB  CC        VV    VV       TT      QQ    QQ   FF
CC         00    00  BB      BB  CC        VV    VV       TT      QQ    QQ   FF
CC         00    00  BB      BB  CC        VV    VV       TT      QQ    QQ   FF
CC         00    00  BBBBBBBB    CC        VV    VV       TT      QQ    QQ   FFFFFFF
CC         00    00  BBBBBBBB    CC        VV    VV       TT      QQ    QQ   FFFFFFF
CC         00    00  BB      BB  CC        VV    VV       TT      QQ  QQ QQ  FF
CC         00    00  BB      BB  CC         VV  VV        TT      QQ  QQ QQ  FF
CC         00    00  BB      BB  CC         VV  VV        TT      QQ    QQ   FF      ....
CC         00    00  BB      BB  CC          VV VV        TT      QQ    QQ   FF      ....
CCCCCCCC    000000   BBBBBBBB    CCCCCCC     VV VV        TT      QQQQ  QQ   FF      ....
CCCCCCCC    000000   BBBBBBBB    CCCCCCC     VV           TT      QQQQ  QQ   FF      ....
```

```
LL          IIIIII   SSSSSSSS
LL          IIIIII   SSSSSSSS
LL            II     SS
LL            II     SS
LL            II     SS
LL            II     SS
LL            II       SSSSSS
LL            II       SSSSSS
LL            II           SS
LL            II           SS
LL            II           SS
LL            II           SS
LLLLLLLLL   IIIIII   SSSSSSSS
LLLLLLLLL   IIIIII   SSSSSSSS
```

COB$CVTQF_R8
1-004

K 13

COBOL Convert Quad to Floating

15-SEP-1984 23:40:20   VAX/VMS Macro V04-00   Page 1
6-SEP-1984 10:43:29   [COBRTL.SRC]COBCVTQF.MAR;1   (1)

```
0000     1          .TITLE   COB$CVTQF_R8   COBOL Convert Quad to Floating
0000     2          .IDENT   /1-004/        ; File: COBCVTQF.MAR
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   ; FACILITY: COBOL SUPPORT
0000    29   ;++
0000    30   ; ABSTRACT:
0000    31   ;        This module contains the routine that converts quadword numbers
0000    32   ;        to floating.
0000    33   ;
0000    34   ;
0000    35   ;--
0000    36   ;
0000    37   ; VERSION: 1
0000    38   ;
0000    39   ; HISTORY:
0000    40   ;
0000    41   ; AUTHOR:
0000    42   ;        Marty Jack, 14-Mar-1979
0000    43   ;
0000    44   ; MODIFIED BY:
0000    45   ;
0000    46   ;
0000    47   ;
```

COB$CVTQF_R8
1-004

L 13
COBOL Convert Quad to Floating   15-SEP-1984 23:40:20   VAX/VMS Macro V04-00   Page   2
HISTORY ; Detailed Current Edit History 6-SEP-1984 10:43:29   [COBRTL.SRC]COBCVTQF.MAR;1   (2)

```
0000   49              .SBTTL  HISTORY          ; Detailed Current Edit History
0000   50
0000   51
0000   52  ; Edit History for Version 1 of COBCVTQF
0000   53  ;
0000   54  ; 1-001 - Original.  MLJ 14-Mar-1979
0000   55  ; 1-002 - Make external references explicit. RKR 17-JULY-1979
0000   56  ; 1-003 - Change all references to FOR$CNV_IN_DEFG to OTS$CVT_T_D
0000   57  ;          RKR 27-SEPT-79
0000   58  ; 1-004 - Cosmetic changes.  RKR 18-OCT-79
```

```
                        0000    60            .SBTTL   DECLARATIONS
                        0000    61
                        0000    62 ;
                        0000    63 ; INCLUDE FILES:
                        0000    64 ;
                        0000    65        $DSCDEF
                        0000    66
                        0000    67 ;
                        0000    68 ; EXTERNAL SYMBOLS:
                        0000    69
                        0000    70            .DSABL   GBL           ; Prevent undeclared symbols from being
                        0000    71                                   ; automatically global
                        0000    72
                        0000    73            .EXTRN   OTS$CVT_T_D   ; D, E, F, G Format Converison Routine
                        0000    74 ;
                        0000    75
                        0000    76 ;
                        0000    77 ; MACROS:
                        0000    78 ;      NONE
                        0000    79 ;
                        0000    80
                        0000    81 ;
                        0000    82 ; PSECT DECLARATIONS:
                    00000000    83            .PSECT  _COB$CODE     PIC, SHR, LONG, EXE, NOWRT
                        0000    84
                        0000    85 ;
                        0000    86 ; EQUATED SYMBOLS:
                        0000    87 ;      NONE
                        0000    88 ;
                        0000    89
                        0000    90 ;
                        0000    91 ; OWN STORAGE:
                        0000    92 ;
                        0000    93 ;+
                        0000    94 ; The following constant has the value 2**32.  It is used for scaling
                        0000    95 ; the high 32 bits and for compensating for unsigned arithmetic.
                        0000    96 ;-
6C 29 67 49 29 04       0000    97 BIAS:   .PACKED 4294967296     ; 2**32
        0000000A        0006    98 BIAS_DIGITS=10
                        0006    99 ;
```

```
                                      0006    101            .SBTTL  COB$CVTQF_R8
                                      0006    102
                                      0006    103  ;++
                                      0006    104  ; FUNCTIONAL DESCRIPTION:
                                      0006    105  ;
                                      0006    106  ;       Converts 64-bit (quadword) numbers to floating.
                                      0006    107  ;
                                      0006    108  ; CALLING SEQUENCE:
                                      0006    109  ;
                                      0006    110  ;       JSB COB$CVTQF_R8 (scale.rl.v, src.rq.r, dst.wf.r)
                                      0006    111  ;
                                      0006    112  ;       Arguments are passed in R6, R7, and R8.
                                      0006    113  ;
                                      0006    114  ; INPUT PARAMETERS:
                                      0006    115  ;
                                      0006    116  ;       SCALE.rl.v              The power of ten by which the internal
                                      0006    117  ;                               representation of the source must be
                                      0006    118  ;                               multiplied to scale the same as the
                                      0006    119  ;                               internal representation of the dest.
                                      0006    120  ;       SRC.rq.r                The number to be converted
                                      0006    121  ;
                                      0006    122  ; IMPLICIT INPUTS:
                                      0006    123  ;
                                      0006    124  ;       All of the trap bits in the PSL are assumed off.
                                      0006    125  ;
                                      0006    126  ; OUTPUT PARAMETERS:
                                      0006    127  ;
                                      0006    128  ;       DST.wf.r                The place to store the converted number
                                      0006    129  ;
                                      0006    130  ; IMPLICIT OUTPUTS:
                                      0006    131  ;
                                      0006    132  ;       NONE
                                      0006    133  ;
                                      0006    134  ; FUNCTION VALUE:
                                      0006    135  ;
                                      0006    136  ;       1 = SUCCESS, 0 = FAILURE
                                      0006    137  ;
                                      0006    138  ; SIDE EFFECTS:
                                      0006    139  ;
                                      0006    140  ;       Destroys registers R0 through R8.
                                      0006    141  ;
                                      0006    142  ;--
                                      0006    143
                                      0006    144
                                      0006    145  COB$CVTQF_R8::
                         5E   28  C2  0006    146            SUBL2   #40,SP                     ; Space for temp string and result
                                      0009    147  ;
                                      0009    148  ; Convert the quadword input to packed.
                                      0009    149  ;
         04 A7  67  01  1F  EC        0009    150            CMPV    #31,#1,(R7),4(R7)          ; Is number in longword range?
                         07   12      000F    151            BNEQ    11$                        ; Br if not to slower code
             08 AE  13   67  F9       0011    152            CVTLP   (R7),#19,8(SP)             ; Convert low order longword
                         20   11      0016    153            BRB     13$                        ; To common code
         6E  0A  04 A7   F9           0018    154  11$:      CVTLP   4(R7),#10,(SP)             ; Convert high order longword
   13  6E  0A  DF AF  0A  25          001D    155            MULP    #BIAS_DIGITS,BIAS,#10,(SP),#19,8(SP)
                         08 AE        0024
                                      0026    156                                              ; Multiply by 2**32
```

B 14

COB$CVTQF_R8   COBOL Convert Quad to Floating   15-SEP-1984 23:40:20 VAX/VMS Macro V04-00   Page 5
1-004   COB$CVTQF_R8   6-SEP-1984 10:43:29 [COBRTL.SRC]COBCVTQF.MAR;1   (4)

```
              6E   0A   67   F9   0026   157              CVTLP    (R7),#10,(SP)             ; Convert low order longword
                        06   18   002A   158              BGEQ     12$                       ; Br if nonnegative
      6E   0A   D0   AF   0A   20   002C   159              ADDP4    #BIAS_DIGITS,BIAS,#10,(SP)
                             C032   160                                                      ; Correct for signed conversion
   08 AE   13   6E   0A   20   0032   161  12$:   ADDP4    #10,(SP),#19,8(SP)        ; Sum low and high order parts
                             0038   162  ;
                             0038   163  ; Convert the packed intermediate to leading separate.
                             0038   164  ;
14 AE   13   08 AE   13   08   0038   165  13$:   CVTPS    #19,8(SP),#19,20(SP)      ; Make a separate sign string
                             003F   166  ;
                             003F   167  ; Make a descriptor for the leading separate string.
                             003F   168  ;
                        53   DD   003F   169              PUSHL    R3                        ; Address = temp string
                   7E   01   90   0041   170              MOVB     #DSC$K_CLASS_S,-(SP)      ; Class = static
                   7E   0E   90   0044   171              MOVB     #DSC$K_DTYPE_T,-(SP)      ; Data type = ASCII text
                   7E   14   B0   0047   172              MOVW     #20,-(SP)                 ; Length = 20 bytes
                             004A   173  ;
                             004A   174  ; Now call the conversion routine.
                             004A   175  ;
                   7E   56   CE   004A   176              MNEGL    R6,-(SP)                  ; Scale factor
                        00   DD   004D   177              PUSHL    #0                        ; Digits in fraction
                   10 AE   9F   004F   178              PUSHAB   16(SP)                    ; Address of result area
                   0C AE   9F   0052   179              PUSHAB   12(SP)                    ; Address of descriptor
   00000000'GF   04   FB   0055   180              CALLS    #4,G^OTS$CVT_T_D          ; Call the routine
                   0D 50   E9   005C   181              BLBC     R0,15$                    ; Failure, must be overflow
              68   08 AE   76   005F   182              CVTDF    8(SP),(R8)                ; Store result
                        07   1D   0063   183              BVS      15$                       ; Br if overflowed
                   50   01   D0   0065   184              MOVL     #1,R0                     ; Indicate success
                   5E   30   C0   0068   185  14$:   ADDL2    #48,SP                    ; Delete stack temps
                        05   006B   186              RSB                                ; Return
                             006C   187  ;
                             006C   188  ; Come here on overflow to store the reserved operand.
                             006C   189  ;
              68   01   0F   78   006C   190  15$:   ASHL     #15,#1,(R8)               ; Store reserved operand
                   50   D4   0070   191              CLRL     R0                        ; Indicate failure
                   F4   11   0072   192              BRB      14$                       ; Delete stack temps and return
                             0074   193  ;
                             0074   194              .END
```

```
BIAS                            00000000 R     02
BIAS_DIGITS                   = 0000000A
COB$CVTQF_R8                    00000006 RG    02
DSC$K_CLASS_S                 = 00000001
DSC$K_DTYPE_T                 = 0000000E
OTS$CVT_T_D                     ********  X    00
```

```
                              +------------------+
                              ! Psect synopsis !
                              +------------------+
```

```
PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000 (    0.)   00 (  0.)  NOPIC  USR  CON  ABS  LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                           00000000 (    0.)   01 (  1.)  NOPIC  USR  CON  ABS  LCL NOSHR  EXE   RD    WRT NOVEC BYTE
_COB$CODE                       00000074 (  116.)   02 (  2.)    PIC  USR  CON  REL  LCL  SHR   EXE   RD  NOWRT NOVEC LONG
```

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

```
Phase                   Page faults    CPU Time        Elapsed Time
-----                   -----------    --------        ------------
Initialization                  30     00:00:00.06     00:00:01.16
Command processing             119     00:00:00.29     00:00:02.78
Pass 1                         139     00:00:01.17     00:00:05.83
Symbol table sort                0     00:00:00.11     00:00:00.56
Pass 2                          50     00:00:00.39     00:00:02.51
Symbol table output              2     00:00:00.01     00:00:00.01
Psect synopsis output            2     00:00:00.01     00:00:00.01
Cross-reference output           0     00:00:00.00     00:00:00.00
Assembler run totals           344     00:00:02.05     00:00:12.89
```

The working set limit was 1050 pages.
8737 bytes (18 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 135 non-local and 5 local symbols.
194 source lines were read in Pass 1, producing 10 object records in Pass 2.
8 pages of virtual memory were used to define 7 macros.

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

```
Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2               4
```

190 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

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