```
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        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
```

```
LL              IIIIII    BBBBBBBB    EEEEEEEEEE  MM      MM    OOOOOO    DDDDDDDD    HH      HH
LL              IIIIII    BBBBBBBB    EEEEEEEEEE  MM      MM    OOOOOO    DDDDDDDD    HH      HH
LL                II      BB      BB  EE          MMMM  MMMM  OO      OO  DD      DD  HH      HH
LL                II      BB      BB  EE          MMMM  MMMM  OO      OO  DD      DD  HH      HH
LL                II      BB      BB  EE          MM  MM  MM  OO      OO  DD      DD  HH      HH
LL                II      BB      BB  EE          MM  MM  MM  OO      OO  DD      DD  HH      HH
LL                II      BBBBBBBB    EEEEEEEE    MM      MM  OO      OO  DD      DD  HHHHHHHHHH
LL                II      BBBBBBBB    EEEEEEEE    MM      MM  OO      OO  DD      DD  HHHHHHHHHH
LL                II      BB      BB  EE          MM      MM  OO      OO  DD      DD  HH      HH
LL                II      BB      BB  EE          MM      MM  OO      OO  DD      DD  HH      HH
LL                II      BB      BB  EE          MM      MM  OO      OO  DD      DD  HH      HH      ....
LL                II      BB      BB  EE          MM      MM  OO      OO  DD      DD  HH      HH      ....
LLLLLLLLLL      IIIIII    BBBBBBBB    EEEEEEEEEE  MM      MM    OOOOOO    DDDDDDDD    HH      HH      ....
LLLLLLLLLL      IIIIII    BBBBBBBB    EEEEEEEEEE  MM      MM    OOOOOO    DDDDDDDD    HH      HH      ....

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

10

Table of contents

```
0000     1          .TITLE  LIB$EMODH - Extended multiply and integerize H
0000     2          .IDENT  /1-002/         ; File: LIBEMODH.MAR  Edit: SBL1002
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY      *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    19 ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: General Utility Library
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       Extend precision of multiplier, multiply by multiplicand
0000    35 ;       and extract integer and fractional portion of result.
0000    36 ;
0000    37 ; ENVIRONMENT: User Mode, AST Reentrant
0000    38 ;
0000    39 ;--
0000    40 ; AUTHOR: Steven B. Lionel, CREATION DATE: 31-July-1979
0000    41 ;
0000    42 ; MODIFIED BY:
0000    43 ;
```

```
0000     45          .SBTTL  Edit History
0000     46 ; 1-001 - Original.  SBL 31-July-1979
0000     47 ; 1-002 - Use local handler to insure that exceptions other than those documented
0000     48 ;         as statuses are resignalled.  SBL 25-Sept-1980
0000     49 ;
```

```
                     0000     51          .SBTTL  DECLARATIONS
                     0000     52 ;
                     0000     53 ; INCLUDE FILES:
                     0000     54 ;
                     0000     55          $CHFDEF                         ; Condition handling macros
                     0000     56          $SSDEF                          ; System symbol definitions
                     0000     57 ;
                     0000     58 ; EXTERNAL SYMBOLS:
                     0000     59 ;
                     0000     60
                     0000     61          .EXTRN  LIB$SIG_TO_RET          ; Library routine to convert a signal
                     0000     62                                          ; to error return to caller
                     0000     63                                          ; of LIB$EMODH.
                     0000     64                                          ; R0 = signaled condition
                     0000     65
                     0000     66
                     0000     67 ;
                     0000     68 ; MACROS:
                     0000     69 ;
                     0000     70
                     0000     71 ;
                     0000     72 ; EQUATED SYMBOLS:
                     0000     73 ;
                     0000     74
00000004             0000     75          mulr = 4                        ; multiplier
00000008             0000     76          mulrx = 8                       ; multiplier extension
0000000C             0000     77          muld = 12                       ; multiplicand
00000010             0000     78          int = 16                        ; integer portion returned
00000014             0000     79          fract = 20                      ; fractional portion returned
                     0000     80
                     0000     81 ;
                     0000     82 ; OWN STORAGE:
                     0000     83 ;
                     0000     84
                     0000     85 ;
                     0000     86 ; PSECT DECLARATIONS:
                     0000     87 ;
                 00000000     88          .PSECT _LIB$CODE        PIC, USR, CON, REL, LCL, SHR, -
                     0000     89                                  EXE, RD, NOWRT, LONG
                     0000     90
```

```
                        .SBTTL  LIB$EMODH  - Extended multiply and integerize
                ;++
                ; FUNCTIONAL DESCRIPTION:
                ;
                ;       LIB$EMODH provides the functionality of the VAX hardware
                ;       instruction EMODH to high-level language users.
                ;
                ;       The floating point multiplier extension operand (second operand)
                ;       is concatenated with the floating point multiplier (first
                ;       operand) to gain 15 additional low order fraction bits.
                ;       The multiplicand operand is multiplied by the extended
                ;       multiplier operand.  After multiplication, the integer
                ;       portion is extracted and a 128 bit floating point number is
                ;       formed from the fractional part of the product by truncating
                ;       extra bits.  The multiplication is such that the result is
                ;       eguivalent to the exact product truncated to a fraction
                ;       field of 128 bits.  Regarding the result as the sum of an
                ;       integer and fraction of the same sign, the integer operand
                ;       is replaced by the integer part of the result and the
                ;       fraction operand is replaced by the rounded fractional
                ;       part of the result.
                ;
                ; CALLING SEQUENCE:
                ;
                ;       status.wlc.v = LIB$EMODH (mulr.rh.r, mulrx.rw.r, muld.rh.r,
                ;                                 int.wl.r, fract.wh.r)
                ;
                ; INPUT PARAMETERS:
                ;
                ;       mulr.rh.r       - floating point multiplier
                ;       mulrx.rw.r      - word to be appended to multiplier fraction
                ;       muld.rh.r       - floating point multiplicand
                ;
                ; IMPLICIT INPUTS:
                ;
                ;       NONE
                ;
                ; OUTPUT PARAMETERS:
                ;
                ;       int.wl.r        - integer portion of result
                ;       fract.wh.r      - fractional portion of result
                ;
                ; IMPLICIT OUTPUTS:
                ;
                ;       NONE
                ;
                ; FUNCTION VALUE:
                ;
                ;       SS$_NORMAL      - successful execution
                ;       SS$_INTOVF      - integer overflow or floating overflow
                ;       SS$_FLTUND      - floating underflow
                ;       SS$_ROPRAND     - reserved operand
                ;
                ; SIDE EFFECTS:
                ;
                ;       All other exceptions are signalled.
                ;
```

```
                                0000   149 ;--
                                0000   150
                         4000   0000   151         .ENTRY LIB$EMODH, ^M<IV>                ; Entry point
                                0002   152
                6D   16'AF   9E 0002   153         MOVAB   B^HANDLER, (FP)                 ; Enable local handler to
                                0006   154                                                 ; process exceptions
                                0006   155
10 BC   0C BC   08 BC   04 BC 74FD 0006   156         EMODH   @mulr(AP), -                    ; perform multiplication
                       14 BC    0010
                                0012   157                 @mulrx(AP), -                   ; trap on exception to
                                0012   158                 @muld(AP), -                    ; handler which will
                                0012   159                 @int(AP), -                     ; unwind a return error
                                0012   160                 @fract(AP)                      ; condition in R0 to
                                0012   161                                                 ; caller of LIB$EMODH.
                                0012   162
                   50   01   9A 0012   163         MOVZBL  #1, R0                          ; success status code
                                0015   164
                         04     0015   165         RET                                     ; return
                                0016   166
                                0016   167 HANDLER:
                         0000   0016   168         .WORD   0
                                0018   169
                                0018   170 ;+
                                0018   171 ;   If the exception is one of the documented exceptions for this routine,
                                0018   172 ;   call LIB$SIG_TO_RET to return it as a status.  Otherwise, resignal.
                                0018   173 ;   Also, resignal if the depth is not zero.
                                0018   174 ;-
                                0018   175
                50   08 AC   D0 0018   176         MOVL    CHF$L_MCHARGLST(AP), R0         ; Get mechanism vector address
                     08 A0   D5 001C   177         TSTL    CHF$L_MCH_DEPTH(R0)             ; Is depth zero?
                        32   12 001F   178         BNEQ    90$                             ; If not, resignal
                51   04 AC   D0 0021   179         MOVL    CHF$L_SIGARGLST(AP), R1         ; Get signal vector address
                50   04 A1   D0 0025   180         MOVL    CHF$L_SIG_NAME(R1), R0          ; Get signalled condition
            047C 8F     50   B1 0029   181         CMPW    R0, #SS$_INTOVF                 ; Compare conditions
                        1B   13 002E   182         BEQL    10$                             ; If it matches, don't resignal
            049C 8F     50   B1 0030   183         CMPW    R0, #SS$_FLTUND
                        14   13 0035   184         BEQL    10$
            0454 8F     50   B1 0037   185         CMPW    R0, #SS$_ROPRAND
                        0D   13 003C   186         BEQL    10$
            04C4 8F     50   B1 003E   187         CMPW    R0, #SS$_FLTUND_F
                        0E   12 0043   188         BNEQ    90$
 04 A1      049C 8F     3C   FA 0045   189         MOVZWL  #SS$_FLTUND, CHF$L_SIG_NAME(R1) ; Change fault code to trap code
00000000'GF     6C   FA 004B   190 10$:    CALLG   (AP), G^LIB$SIG_TO_RET                  ; Return signal as a status
                     04 0052   191         RET
   50      0918 8F     3C 0053   192 90$:    MOVZWL  #SS$_RESIGNAL, R0                       ; Resignal condition
                     04 0058   193         RET
                        0059   194
                        0059   195
                        0059   196         .END
```

```
CHF$L_MCHARGLST                 = 00000008
CHF$L_MCH_DEPTH                 = 00000008
CHF$L_SIGARGLST                 = 00000004
CHF$L_SIG_NAME                  = 00000004
FRACT                           = 00000014
HANDLER                           00000016 R      02
INT                             = 00000010
LIB$EMODH                         00000000 RG     02
LIB$SIG_TO_RET                    ******** **   X  00
MULD                            = 0000000C
MULR                            = 00000004
MULRX                           = 00000008
SS$_FLTUND                      = 0000049C
SS$_FLTUND_F                    = 000004C4
SS$_INTOVF                      = 0000047C
SS$_RESIGNAL                    = 00000918
SS$_ROPRAND                     = 00000454
```

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
_LIB$CODE	00000059 (89.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.03	00:00:01.67
Command processing	111	00:00:00.30	00:00:02.18
Pass 1	190	00:00:02.72	00:00:09.17
Symbol table sort	0	00:00:00.42	00:00:01.16
Pass 2	51	00:00:00.58	00:00:01.34
Symbol table output	4	00:00:00.02	00:00:00.02
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	390	00:00:04.08	00:00:15.55

The working set limit was 1200 pages.
21526 bytes (43 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 428 non-local and 2 local symbols.
196 source lines were read in Pass 1, producing 13 object records in Pass 2.
9 pages of virtual memory were used to define 8 macros.

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

Macro library name                    Macros defined
------------------                    --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                5

486 GETS were required to define 5 macros.

There were no errors, warnings or information messages.

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

LIBEMODH
LIS

LIBEMODD
LIS

LIBESTEMU
LIS

LIBEF
LIS

LIBEMULAT
LIS

LIBFFS
LIS

LIBFINCVT
LIS

LIBE2AREV
LIS

LIBFAO
LIS

LIBEMODG
LIS

LIBEXTV
LIS

LIBEDIV
LIS

LIBESTABL
LIS

LIBFFC
LIS

LIBEMODF
LIS

LIBEMUL
LIS

LIBFILSCA
LIS

LIBEXTZV
LIS

LIBEBCASC
LIS

LIBFAOL
LIS