```
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
   SSSSSSSSSSS  UUU          UUU  MMM          MMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMMMMMM    MMMMMM
SSS             UUU          UUU  MMM MMM MMM MMM
SSS             UUU          UUU  MMM  MMM  MMM MMM
SSS             UUU          UUU  MMM   M   MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
   SSSSSSSSS    UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
            SSS UUU          UUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
SSSSSSSSSSS     UUUUUUUUUUUUUUUU  MMM       MMM MMM
```

```
  SSSSSSSS  UU      UU  MM      MM  EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    RRRRRRRR
  SSSSSSSS  UU      UU  MM      MM  EEEEEEEEEE  RRRRRRRR    RRRRRRRR      000000    RRRRRRRR
SS          UU      UU  MMMM  MMMM  EE          RR      RR  RR      RR  00      00  RR      RR
SS          UU      UU  MMMM  MMMM  EE          RR      RR  RR      RR  00      00  RR      RR
SS          UU      UU  MM  MM  MM  EE          RR      RR  RR      RR  00      00  RR      RR
SS          UU      UU  MM  MM  MM  EE          RR      RR  RR      RR  00      00  RR      RR
  SSSSSS    UU      UU  MM      MM  EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  RRRRRRRR
  SSSSSS    UU      UU  MM      MM  EEEEEEEE    RRRRRRRR    RRRRRRRR    00      00  RRRRRRRR
        SS  UU      UU  MM      MM  EE          RR  RR      RR  RR      00      00  RR  RR
        SS  UU      UU  MM      MM  EE          RR  RR      RR  RR      00      00  RR  RR
        SS  UU      UU  MM      MM  EE          RR    RR    RR    RR    00      00  RR    RR        ....
        SS  UU      UU  MM      MM  EE          RR    RR    RR    RR    00      00  RR    RR        ....
SSSSSSSS    UUUUUUUUUU  MM      MM  EEEEEEEEEE  RR      RR  RR      RR    000000    RR      RR      ....
SSSSSSSS    UUUUUUUUUU  MM      MM  EEEEEEEEEE  RR      RR  RR      RR    000000    RR      RR      ....

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

Table of contents

```
0000     1          .TITLE  SUM$ERROR
0000     2          .IDENT  'V04-000'
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:
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ;       Source Update Merge error processing
0000    35  ;
0000    36  ; ENVIRONMENT:  USER MODE
0000    37  ;
0000    38  ; AUTHOR:       R. Newland
0000    39  ;
0000    40  ; MODIFIED BY:
0000    41  ;
0000    42  ;       V03-002 MTR0001         Mike Rhodes             19-Jan-1983
0000    43  ;               Add routine SUM$SET_SIGNAL which allows a user to select
0000    44  ;               SUMSHR's method of Issuing diagnostics $PUTMSG or SIGNAL.
0000    45  ;               The routines PUT_MSG and SUM$LIB_ERR have been modified
0000    46  ;               to use this selection critirea ($PUTMSG is the default).
0000    47  ;
0000    48  ;       V03-001 BLS0173         Benn Schreiber          26-MAY-1982
0000    49  ;               Make error routines into entry points so they can be vectored.
0000    50  ;
0000    51  ;       V003    TMH0003         Tim Halvorsen           26-Dec-1981
0000    52  ;               Fix error reporting, which was broken because it relied
0000    53  ;               on the number of arguments to PUTMSG to point to one of
0000    54  ;               the arguments - now broken because PUTMSG has an extra
0000    55  ;               optional argument.
0000    56  ;
0000    57  ;       V02-002 CNH0037         Chris Hume              21-Sep-1980
```

```
0000     58 ;                Added SUM$WRITE_ERR while obsoleting SUMSLP's ERROR.MAR
0000     59 ;
0000     60 ;        V02-001                 B. Schreiber            21-Mar-1980
0000     61 ;                        Make totally position independent
0000     62 ;--
```

```
                     0000     64          .SBTTL  DECLARATIONS
                     0000     65 ;
                     0000     66 ;
                     0000     67 ; Macro definitions
                     0000     68 ;
                     0000     69          $FABDEF                         ; FAB block
                     0000     70          $NAMDEF                         ; NAM block
                     0000     71          $RABDEF                         ; RAB block
                     0000     72          $SHRDEF                         ; Shared messages
                     0000     73          $STSDEF                         ; Status values
                     0000     74 ;
                     0000     75 ;
                     0000     76 ;
                     0000     77 ; Local storage
                     0000     78 ;
                     0000     79 ;
                 00000000     80          .PSECT  SUM$RO_DATA,NOEXE,NOWRT,LONG
                     0000     81 ;
                     0000     82 ;
                     0000     83 SUM_NAME:
 4D 55 53 00'        0000     84          .ASCIC  /SUM/                   ; Facility name for messages
             03      0000
                     0004     85 ;
                     0004     86 ;
                 00000000     87          .PSECT  SUM$RW_DATA,NOEXE,LONG
                     0000     88 ;
                     0000     89 ;
                     0000     90 SUM_CONTEXT:
                00   0000     91          .BYTE   0                       ; Context bits...Low bit indicates
                     0001     92                                          ; diagnostics issued by $PUTMSG (0 - default
                     0001     93                                          ;       or              SIGNAL  (1)
```

```
                                0001     95          .SBTTL  RMS ERROR PROCESSING
                                0001     96 ;
                                0001     97 ;
                                0001     98 ;++
                                0001     99 ; Functional description:
                                0001    100 ;
                                0001    101 ;       These routines are called as RMS error processing action routines.
                                0001    102 ;       They form and print a message describing the error.
                                0001    103 ;
                                0001    104 ; Inputs:
                                0001    105 ;
                                0001    106 ;       4(AP) = Address of RMS control block (either FAB or RAB)
                                0001    107 ;
                                0001    108 ; Outputs:
                                0001    109 ;
                                0001    110 ;       None
                                0001    111 ;
                                0001    112 ;--
                                0001    113 ;
                           00000000     114          .PSECT  SUM$CODE,NOWRT,LONG
                                0000    115 ;
                           003C 0000    116          .ENTRY  SUM$OPEN_ERR,^M<R2,R3,R4,R5>
                        4B      10 0002 117          BSBB    FAB_ERR                 ; Process for FAB control block
    51  0002109A            8F  D0 0004 118          MOVL    #SHR$_OPENIN!STS$K_ERROR!<2@16>,R1 ; Get shared message code
    00000000'8F             54  D1 000B 119          CMPL    R4,#RMS$_FNF            ; Was error 'file not found'?
                            05  12 0012 120          BNEQ    10$                     ; No if NEQL
51  03  00  03              F0  0014 121          INSV    #STS$K_INFO, -          ; Yes so reduce severity to information
                                0019 122                  #STS$V_SEVERITY, -
                                0019 123                  #STS$S_SEVERITY,R1
                                0019 124 10$:
                        6B      10 0019 125          BSBB    PUT_MSG                 ; Put message
                            04  10 001B 126          RET
                                001C 127 ;
                                001C 128 ;
                           003C 001C 129          .ENTRY  SUM$CLOSE_ERR,^M<R2,R3,R4,R5>
                        2F      10 001E 130          BSBB    FAB_ERR                 ; Process for FAB control block
51  00021052            8F  D0 0020 131          MOVL    #SHR$_CLOSEIN!STS$K_ERROR!<2@16>,R1 ; Get shared message code
                        5D      10 0027 132          BSBB    PUT_MSG                 ; Put message
                            04  10 0029 133          RET
                                002A 134 ;
                                002A 135 ;
                           003C 002A 136          .ENTRY  SUM$READ_ERR,^M<R2,R3,R4,R5>
                        2B      10 002C 137          BSBB    RAB_ERR                 ; Process for RAB control block
    00000000'8F             54  D1 002E 138          CMPL    R4,#RMS$_EOF            ; Is error end-of-file?
                            09  13 0035 139          BEQL    10$                     ; Yes if EQL, so don't report
51  000210B2                8F  D0 0037 140          MOVL    #SHR$_READERR!STS$K_ERROR!<2@16>,R1 ; Get shared message code
                        46      10 003E 141          BSBB    PUT_MSG                 ; Put message
                                0040 142 10$:
                            04  10 0040 143          RET
                                0041 144 ;
                                0041 145 ;
                           003C 0041 146          .ENTRY  SUM$WRITE_ERR,^M<R2,R3,R4,R5>
                        14      10 0043 147          BSBB    RAB_ERR                 ; Process for RAB control block
51  000210D2                8F  D0 0045 148          MOVL    #SHR$_WRITEERR!STS$K_ERROR!<2@16>,R1 ; Get shared message code
                        38      10 004C 149          BSBB    PUT_MSG                 ; Put message
                            04  10 004E 150          RET
```

```
                     004F    152          .SBTTL  FAB_ERR and RAB_ERR
                     004F    153 ;
                     004F    154 ;
                     004F    155 ;++
                     004F    156 ; Functional description:
                     004F    157 ;
                     004F    158 ;       These two routines are called to form the data required to
                     004F    159 ;       use the system PUTMSG routine.  Separate entry points exist
                     004F    160 ;       for the RMS control block being an FAB or RAB
                     004F    161 ;
                     004F    162 ; Inputs:
                     004F    163 ;
                     004F    164 ;       4(AP) = Address of RMS control block
                     004F    165 ;
                     004F    166 ; Outputs:
                     004F    167 ;       R0 = FAB address
                     004F    168 ;       R1 = NAM block address
                     004F    169 ;       R2 = Size of file spec string
                     004F    170 ;       R3 = Address of file spec string
                     004F    171 ;       R4 = Completion status code
                     004F    172 ;       R5 = Status value
                     004F    173 ;
                     004F    174 ;--
                     004F    175          .ENABL  LSB
                     004F    176 ;
                     004F    177 FAB_ERR:
50   04 AC    D0     004F    178          MOVL    4(AP),R0                ; Get FAB address
54   08 A0    7D     0053    179          MOVQ    FAB$L_STS(R0),R4        ; Get completion code and status value
      0C    11     0057    180          BRB     10$
                     0059    181 ;
                     0059    182 RAB_ERR:
51   04 AC    D0     0059    183          MOVL    4(AP),R1                ; Get RAB address
50   3C A1    D0     005D    184          MOVL    RAB$L_FAB(R1),R0        ; and then FAB address
54   08 A1    7D     0061    185          MOVQ    RAB$L_STS(R1),R4        ; Get completion code and status value
                     0065    186 10$:
51   28 A0    D0     0065    187          MOVL    FAB$L_NAM(R0),R1        ; Get NAM block address
53   04 A1    D0     0069    188          MOVL    NAM$L_RSA(R1),R3        ; Get resultant string address
52   03 A1    9A     006D    189          MOVZBL  NAM$B_RSL(R1),R2        ; and length
      12    12     0071    190          BNEQ    20$                     ; If NEQ resultant string was formed
53   0C A1    D0     0073    191          MOVL    NAM$L_ESA(R1),R3        ; Get expanded string address
52   0B A1    9A     0077    192          MOVZBL  NAM$B_ESL(R1),R2        ; and length
      08    12     007B    193          BNEQ    20$                     ; If NEQ expanded string was formed
53   2C A0    D0     007D    194          MOVL    FAB$L_FNA(R0),R3        ; Get input file name
52   34 A0    9A     0081    195          MOVZBL  FAB$B_FNS(R0),R2        ; and size
                     0085    196 20$:
      05    05     0085    197          RSB
                     0086    198 ;
                     0086    199 ;
                     0086    200          .DS BL  LSB
```

```
                     0086   202          .SBTTL  PUT_MSG
                     0086   203 ;
                     0086   204 ;
                     0086   205 ;++
                     0086   206 ; Functional description:
                     0086   207 ;
                     0086   208 ;       This routine takes message data and calls the system service
                     0086   209 ;       to expand and put the message to the processes SYS$OUTPUT stream
                     0086   210 ;
                     0086   211 ; Inputs:
                     0086   212 ;
                     0086   213 ;       R1 = Message code
                     0086   214 ;       R2 = Size of file spec string
                     0086   215 ;       R3 = Address of file spec string
                     0086   216 ;       R4 = RMS completion status code
                     0086   217 ;       R5 = RMS status value
                     0086   218 ;
                     0086   219 ; Outputs:
                     0086   220 ;
                     0086   221 ;       None
                     0086   222 ;
                     0086   223 ;--
                     0086   224 ;
                     0086   225 PUT_MSG:
11 00000000'EF   E8  0086   226          BLBS    SUM_CONTEXT, 10$        ; Diagnostics issued from $PUTMSG or SIGNAL?
   00000000'EF   DF  008D   227          PUSHAL  SUM_NAME                ; Form string descriptor for facility name
    50    00 BE   9A  0093   228          MOVZBL  @(SP),R0                ; Get length of string
                6E   D6  0097   229          INCL    (SP)                    ; Point to string
                50   DD  0099   230          PUSHL   R0                      ; Form complete descriptor
             50   5E   D0  009B   231          MOVL    SP,R0                   ; Save descriptor address
                0C   BB  009E   232 10$:     PUSHR   #^M<R2,R3>              ; Form string descriptor from R2 and R3
                30   BB  00A0   233          PUSHR   #^M<R4,R5>              ; and push RMS code and status values
              08 AE   7F  00A2   234          PUSHAQ  8(SP)                   ; Address of string descriptor
                01   DD  00A5   235          PUSHL   #1                      ; Number of arguments
                51   DD  00A7   236          PUSHL   R1                      ; Message code
19 00000000'EF   E8  00A9   237          BLBS    SUM_CONTEXT, 20$        ; Diagnostics issued from $PUTMSG or SIGNAL?
                05   DD  00B0   238          PUSHL   #5                      ; Size of message packet
             51   5E   D0  00B2   239          MOVL    SP,R1                   ; Point to message argument vector
                          00B5   240          $PUTMSG_S MSGVEC=(R1),-         ; Convert and print message
                          00B5   241                    FACNAM=(R0)
             5E   28   C0  00C4   242          ADDL2   #10*4,SP                ; Remove 10 longwords from stack
                0A   11  00C7   243          BRB     30$                     ; And return.
00000000'GF   05   FB  00C9   244 20$:     CALLS   #5, G^LIB$SIGNAL        ; Ship the message
             5E   08   C0  00D0   245          ADDL2   #2*4, SP                ; Clean up the stack
                05   00D3   246 30$:     RSB
```

```
                     00D4   248          .SBTTL  LIBRARY ROUTINE ERROR
                     00D4   249 ;
                     00D4   250 ;
                     00D4   251 ; ++
                     00D4   252 ; Functional description:
                     00D4   253 ;
                     00D4   254 ;       This procedure is called to form and output an error message
                     00D4   255 ;       for a library routine error
                     00D4   256 ;
                     00D4   257 ;
                     00D4   258 ; Inputs:
                     00D4   259 ;
                     00D4   260 ;       R0 = Error code
                     00D4   261 ;
                     00D4   262 ;
                     00D4   263 ; Outputs:
                     00D4   264 ;
                     00D4   265 ;       None
                     00D4   266 ;--
                     00D4   267 ;
                     00D4   268 ;
                000C 00D4   269          .ENTRY  SUM$LIB_ERR,^M<R2,R3>
  2C 00000000'EF     E8 00D6   270          BLBS    SUM_CONTEXT, 10$         ; Diagnostics issued from $PUTMSG or SIGNAL?
52  00000000'EF     9E 00DD   271          MOVAB   SUM_NAME,R2              ; Form descriptor for facility name
        53   82     9A 00E4   272          MOVZBL  (R2)+,R3                 ; Get length of name
        7E   53     7D 00E7   273          MOVQ    R3,-(SP)                 ; Stack descriptor
        53   5E     D0 00EA   274          MOVL    SP,R3                    ; And save it's address
             50     DD 00ED   275          PUSHL   R0                       ; Form message arguments on stack
             01     DD 00EF   276          PUSHL   #1                       ; Set size of vector
        50   5E     D0 00F1   277          MOVL    SP,R0                    ; Point to message argument vector
                       00F4   278          $PUTMSG_S MSGVEC=(R0),-          ; Form and output message
                       00F4   279                    FACNAM=(R3)
     50    04 AE     D0 0103   280          MOVL    4(SP),R0                 ; Restore R0
             0F     11 0107   281          BRB     20$                      ; And exit...
             50     DD 0109   282 10$:     PUSHL   R0                       ; Preserve R0 across the call
             00     DD 010B   283          PUSHL   #0                       ; No additional arguments
             50     DD 010D   284          PUSHL   R0                       ; Stack the error code
00000000'GF  02     FB 010F   285          CALLS   #2, G^LIB$SIGNAL         ; Issue the message
             01     BA 0116   286          POPR    #^M<R0>                  ; Restore R0
                    04 0118   287 20$:     RET
```

```
                     0119   289          .SBTTL  SUM$SET_SIGNAL
                     0119   290 ;
                     0119   291 ;
                     0119   292 ;++
                     0119   293 ; Functional Description:
                     0119   294 ;
                     0119   295 ;       This procedure selects SUMSHR's method for issuing
                     0119   296 ;       diagnostic messages.
                     0119   297 ;
                     0119   298 ; Inputs:
                     0119   299 ;
                     0119   300 ;       4(AP)           Value   The low order bit of this value selects
                     0119   301 ;                               0 $PUTMSG (default)
                     0119   302 ;                               1 SIGNAL
                     0119   303 ;
                     0119   304 ; Implicit Inputs:
                     0119   305 ;
                     0119   306 ;       SUM_CONTEXT     Byte    Bit vector for context selection.
                     0119   307 ;
                     0119   308 ; Implicit Outputs:
                     0119   309 ;
                     0119   310 ;       SUM_CONTEXT     Byte    The low order bit is toggled.
                     0119   311 ;
                     0119   312 ;--
                     0119   313 ;
                     0119   314 ;
                0000 0119   315          .ENTRY  SUM$SET_SIGNAL, ^M<>
00000000'EF  01  00  04 AC   F0 011B   316          INSV    4(AP), #0, #1, SUM_CONTEXT      ; Make the selection.
                       04   0125   317          RET
                       04   0126   318          RET
                            0127   319 ;
                            0127   320          .END
```

```
FAB$B_FNS                       =  00000034
FAB$L_FNA                       =  0000002C
FAB$L_NAM                       =  00000028
FAB$L_STS                       =  00000008
FAB_ERR                            0000004F  R       04
LIB$SIGNAL                         ********    X     04
NAM$B_ESL                       =  0000000B
NAM$B_RSL                       =  00000003
NAM$L_ESA                       =  0000000C
NAM$L_RSA                       =  00000004
PUT_MSG                            00000086  R       04
RAB$L_FAB                       =  0000003C
RAB$L_STS                       =  00000008
RAB_ERR                            00000059  R       04
RMS$_EOF                           ********    X     04
RMS$_FNF                           ********    X     04
SHR$_CLOSEIN                    =  00001050
SHR$_OPENIN                     =  00001098
SHR$_READERR                    =  000010B0
SHR$_WRITEERR                   =  000010D0
STS$K_ERROR                     =  00000002
STS$K_INFO                      =  00000003
STS$S_SEVERITY                  =  00000003
STS$V_SEVERITY                  =  00000000
SUM$CLOSE_ERR                      0000001C  RG      04
SUM$LIB_ERR                        000000D4  RG      04
SUM$OPEN_ERR                       00000000  RG      04
SUM$READ_ERR                       0000002A  RG      04
SUM$SET_SIGNAL                     00000119  RG      04
SUM$WRITE_ERR                      00000041  RG      04
SUM_CONTEXT                        00000000  R       03
SUM_NAME                           00000000  R       02
SYS$PUTMSG                         ********  GX      04
```

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
SUM$RO_DATA	00000004	(4.)	02	(2.)	NOPIC	USR	CON	REL	LCL	NOSHR	NOEXE	RD	NOWRT NOVEC LONG
SUM$RW_DATA	00000001	(1.)	03	(3.)	NOPIC	USR	CON	REL	LCL	NOSHR	NOEXE	RD	WRT NOVEC LONG
SUM$CODE	00000127	(295.)	04	(4.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.10	00:00:00.33
Command processing	131	00:00:00.58	00:00:01.88
Pass 1	223	00:00:05.86	00:00:13.32
Symbol table sort	0	00:00:00.66	00:00:01.25
Pass 2	69	00:00:01.29	00:00:02.54

Symbol table output	5	00:00:00.05	00:00:00.05
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	462	00:00:08.57	00:00:19.41

The working set limit was 1050 pages.
33264 bytes (65 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 613 non-local and 9 local symbols.
320 source lines were read in Pass 1, producing 35 object records in Pass 2.
14 pages of virtual memory were used to define 13 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SUM.OBJ]SUM.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	10
TOTALS (all libraries)	10

688 GETS were required to define 10 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:SUMERROR/OBJ=OBJ$:SUMERROR MSRC$:SUMERROR/UPDATE=(ENH$:SUMERROR)+LIB$:SUM/LIB
```

SUMLIST
LIS

SUMMAIN
LIS

SUMTFRVEC
LIS

SUMFILES
LIS

SUMMSG
LIS

SUMERROR
LIS

SUMOPEN
LIS

SYS

UPDATE
LIS

SYS
MAP