M 11

```
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDD         DDD  CCC              LLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

```
                                                                  G  1
**FILE**ID**MESSAGE
```

```
MM      MM  EEEEEEEEEE    SSSSSSSS    SSSSSSSS    AAAAAA      GGGGGGGG  EEEEEEEEEE
MM      MM  EEEEEEEEEE    SSSSSSSS    SSSSSSSS    AAAAAA      GGGGGGGG  EEEEEEEEEE
MMMM  MMMM  EE          SS          SS          AA      AA  GG          EE
MMMM  MMMM  EE          SS          SS          AA      AA  GG          EE
MM  MM  MM  EE          SS          SS          AA      AA  GG          EE
MM  MM  MM  EE          SS          SS          AA      AA  GG          EE
MM      MM  EEEEEEEE      SSSSSS      SSSSSS    AA      AA  GG          EEEEEEEE
MM      MM  EEEEEEEE      SSSSSS      SSSSSS    AA      AA  GG          EEEEEEEE
MM      MM  EE                  SS          SS  AAAAAAAAAA  GG  GGGGGG  EE
MM      MM  EE                  SS          SS  AAAAAAAAAA  GG  GGGGGG  EE
MM      MM  EE                  SS          SS  AA      AA  GG      GG  EE              ....
MM      MM  EE                  SS          SS  AA      AA  GG      GG  EE              ....
MM      MM  EEEEEEEEEE  SSSSSSSS    SSSSSSSS    AA      AA    GGGGGG    EEEEEEEEEE      ....
MM      MM  EEEEEEEEEE  SSSSSSSS    SSSSSSSS    AA      AA    GGGGGG    EEEEEEEEEE      ....

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

```
    (3)       94        OUTPUT ERROR MESSAGE
    (4)       172       FORMAT MESSAGE
    (5)       207       PUTMSG OUTPUT ROUTINE
    (6)       240       OUTPUT FILE MESSAGE OUTPUT (FOR SPAWN)
    (7)       271       OUTPUT FILE MESSAGE OUTPUT
    (8)       305       SET FLUSH RATE COMMAND
    (9)       340       SET THE FLUSH TIMER
    (10)      364       FLUSH OUTPUT FILE
```

```
0000           .TITLE  MESSAGE - MESSAGE OUTPUT ROUTINES
0000           .IDENT  'V04-000'
0000
0000   ;
0000   ;****************************************************************************
0000   ;*                                                                          *
0000   ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000   ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000   ;*  ALL RIGHTS RESERVED.                                                    *
0000   ;*                                                                          *
0000   ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000   ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000   ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000   ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000   ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000   ;*  TRANSFERRED.                                                            *
0000   ;*                                                                          *
0000   ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000   ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000   ;*  CORPORATION.                                                            *
0000   ;*                                                                          *
0000   ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000   ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000   ;*                                                                          *
0000   ;*                                                                          *
0000   ;****************************************************************************
0000   ;
0000   ; MESSAGE OUTPUT ROUTINES
0000   ;
0000   ; D. N. CUTLER 28-MAR-77
0000   ;
0000   ; MODIFIED BY:
0000   ;
0000   ;       V03-010 HWS0065         Harold Schultz          09-May-1984
0000   ;               Use a temporary RAB built on stack when flushing output
0000   ;               to a batch log file instead of current output RAB.
0000   ;
0000   ;       V03-009 HWS0058         Harold Schultz          12-Apr-1984
0000   ;               Add DCL$SPAWNOUT message routine for SPAWN since it
0000   ;               doesn't have valid pointer to work area.(Must be same
0000   ;               as DCL$MSGOUT.
0000   ;
0000   ;       V03-008 PCG0008         Peter George            02-Mar-1984
0000   ;               Move inhibit message bit test.
0000   ;
0000   ;       V03-007 HWS0014         Harold Schultz          21-Feb-1984
0000   ;               If current error number equal to RMS STS value in
0000   ;               PRC_L_STS, signal both the STS and STV messages.
0000   ;
0000   ;       V03-006 PCG0007         Peter George            21-Nov-1983
0000   ;               Fix bug in specifying record for $FLUSH.
0000   ;
0000   ;       V03-005 PCG0006         Peter George            29-Jun-1983
0000   ;               Handle event flags more intelligently.
0000   ;
0000   ;       V03-004 PCG0005         Peter George            24-Jun-1983
0000   ;               Fix broken branch.
```

```
0000     ;
0000     ;        V03-004 PCG0004         Peter George            15-Jun-1983
0000     ;                Do not special case negative statuses.
0000     ;                Return non-zero status when calling SET OUTPUT_RATE
0000     ;                interactively.  Change order of $FLUSH and $SETTIMER
0000     ;                in flush ast routine.
0000     ;
0000     ;        V03-003 PCG0003         Peter George            30-Apr-1983
0000     ;                Add DCL$FLUSH_OUTPUT.
0000     ;                Fix order of args but in $PUTMSG call.
0000     ;
0000     ;        V03-002 PCG0002         Peter George            21-Mar-1983
0000     ;                Add DCL$FORMMSG.
0000     ;
0000     ;        V03-001 PCG0001         Peter George            07-Jan-1983
0000     ;                Use $PUTMSG to output command segment.
0000     ;                Delete DCL$ERROUT.
0000     ;
0000     ;---
```

```
    0000      ;
    0000      ; MACRO LIBRARY CALLS
    0000      ;
    0000
    0000            PRCDEF                          ;DEFINE PROCESS WORK AREA
    0000            WRKDEF                          ;DEFINE COMMAND WORK AREA
    0000            PTRDEF                          ;DEFINE COMMAND TOKENS
    0000            $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
    0000            $IODEF                          ;DEFINE I/O FUNCTION CODES
    0000            $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
    0000            $RABDEF                         ;DEFINE RAB OFFSETS
    0000            $STSDEF                         ;DEFINE STATUS LONGWORD VALUES
    0000            $SFDEF                          ;DEFINE CALL FRAME
    0000
00000000            .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
```

```
                              0000     94         .SBTTL  OUTPUT ERROR MESSAGE
                              0000     95 ;+
                              0000     96 ; DCL$ERRORMSG - OUTPUT ERROR MESSAGE
                              0000     97 ;
                              0000     98 ; THIS ROUTINE IS CALLED TO OUTPUT AN ERROR MESSAGE AND DISPLAY THE SEGMENT
                              0000     99 ; OF THE COMMAND LINE THAT IS IN ERROR.
                              0000    100 ;
                              0000    101 ; INPUTS:
                              0000    102 ;
                              0000    103 ;       R0 = ERROR NUMBER.
                              0000    104 ;       WRK_L_MARKPTR = ADDRESS OF START OF TOKEN IN EXPANSION BUFFER.
                              0000    105 ;       WRK_L_EXPANDPTR = ADDRESS OF NEXT BYTE IN EXPANSION BUFFER.
                              0000    106 ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                              0000    107 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                              0000    108 ;
                              0000    109 ; OUTPUTS:
                              0000    110 ;
                              0000    111 ;       THE APPROPRIATE ERROR MESSAGE IS DISPLAYED ALONG WITH THE SEGMENT OF
                              0000    112 ;       THE COMMAND LINE IN ERROR.
                              0000    113 ;
                              0000    114 ;       R0 IS PRESERVED ACROSS CALL.
                              0000    115 ;-
                              0000    116
                              0000    117 DCL$ERRORMSG::                                  ;OUTPUT ERROR MESSAGE
                 3F     BB    0000    118         PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE REGISTERS
          53 50  1C     E0    0002    119         BBS     #STS$V_INHIB_MSG,R0,60$ ;BR IF NO MESSAGE DESIRED
                              0006    120
                              0006    121 ;
                              0006    122 ; Check if offending text should be output as part of this error message.
                              0006    123 ;
             55  02     D0    0006    124         MOVL    #2,R5                   ;ASSUME NO COMMAND SET WILL BE OUTPUT
             54  0C     D0    0009    125         MOVL    #3*4,R4                 ;  SET ARG COUNT AND STACK USAGE
                 01     E0    000C    126         BBS     #WRK_V_COMMAND,-        ;DO NOT OUTPUT IF COMMAND IN EXECUTION
          22 F0  AA           000E    127                 WRK_Q_FLAGS(R10),40$    ;
              FFEC'     30    0011    128         BSBW    DCL$MARKEDTOKEN         ;GET DESCRIPTOR OF CURRENT PARSE STRING
                 62     95    0014    129         TSTB    (R2)                    ;DOES TOKEN START WITH EOL CHAR?
                 1B     13    0016    130         BEQL    40$                     ;IF SO, ASSUME AT EOL AND SKIP TEXT
                 51     D5    0018    131         TSTL    R1                      ;WILL ANY TOKEN BE SHOWN?
                 17     13    001A    132         BEQL    40$                     ;IF NO ERROR TOKEN, SKIP IT
                              001C    133
                              001C    134 ;
                              001C    135 ; Build the command line part of the message argument vector.
                              001C    136 ;
             7E  51     7D    001C    137 30$:    MOVQ    R1,-(SP)                ;PUSH SEGMENT DESCRIPTOR ON STACK
                 5E     DD    001F    138         PUSHL   SP                      ;PUSH ADDRESS OF SEGMENT DESCRIPTOR
             7E  11     B0    0021    139         MOVW    #^X0011,-(SP)           ;ONLY OUTPUT THE TEXT PART
             7E  01     B0    0024    140         MOVW    #1,-(SP)                ;ONE FAO ARGUMENT
       00038248  8F     DD    0027    141         PUSHL   #CLI$_CMDSEG            ;PUSH MESSAGE CODE
             55  05     D0    002D    142         MOVL    #5,R5                   ;SET ARGUMENT COUNT
             54  20     D0    0030    143         MOVL    #8*4,R4                 ;SET STACK USAGE
                              0033    144
                              0033    145 ;
                              0033    146 ; Build the status part of the message argument vector.
                              0033    147 ;
           0084  CB     D5    0033    148 40$:    TSTL    PRC_L_STS(R11)          ;VALID STS VALUE?
                 12     13    0037    149         BEQL    45$                     ;NO, PROCESS NORMALLY
                              0039    150
          50     0084 CB     D1  0039   151          CMPL    PRC_L_STS(R11),R0          ;STS VALUE = TO CURRENT ERROR NUM?
                      0B     12  003E   152          BNEQ    45$                        ;NO, PROCESS NORMALLY
                                 0040   153
                 02   55     91  0040   154          CMPB    R5,#2                      ;OUTPUT COMMAND SET?
                      06     14  0043   155          BGTR    45$                        ;YES, PROCESS NORMALLY
                 0088 CB     DD  0045   156          PUSHL   PRC_L_STV(R11)             ;PUT STV VALUE INTO MESSAGE VECTOR
                      02     11  0049   157          BRB     50$
                                 004B   158
                      00     DD  004B   159 45$:     PUSHL   #0                         ;CREATE PUTMSG VECTOR (FAO COUNT)
                      50     DD  004D   160 50$:     PUSHL   R0                         ;SET STATUS CODE
                      55     DD  004F   161          PUSHL   R5                         ;# ARGS ON PUTMSG VECTOR
                 50   5E     D0  0051   162          MOVL    SP,R0                      ;ADDRESS OF THE BUFFER DESCRIPTOR
                                 0054   163
                      28     10  0054   164          BSBB    DCL$PUTMSG                 ;SIGNAL PUTMSG VECTOR AT (R0)
                 5E   54     C0  0056   165          ADDL    R4,SP                      ;POP EVERYTHING UP TO BUFFER AND DESC.
                                 0059   166
                                 0059   167 60$:     ASSUME  PRC_L_STV EQ PRC_L_STS+4
                 0084 CB     7C  0059   168          CLRQ    PRC_L_STS(R11)             ;INIT. STS AND STV ERROR NUMBERS
                      3F     BA  005D   169          POPR    #^M<R0,R1,R2,R3,R4,R5>     ;RESTORE REGISTERS
                             05  005F   170          RSB                                ;
```

```
                        0060             .SBTTL  FORMAT MESSAGE
                        0060       ;+
                        0060       ; DCL$FORMMSG - FORMAT MESSAGE
                        0060       ;
                        0060       ; THIS ROUTINE IS CALLED TO CREATE A MESSAGE VECTOR FOR A MESSAG! AND THEN
                        0060       ; CALL DCL$PUTMSG TO OUTPUT THAT MESSAGE.
                        0060       ;
                        0060       ; INPUTS:
                        0060       ;
                        0060       ;         R0 = ERROR NUMBER.
                        0060       ;         R1 = NUMBER OF ARGUMENTS.
                        0060       ;         R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        0060       ;         R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        0060       ;
                        0060       ;         THE FAO ARGUMENTS ARE ON THE STACK, JUST ABOVE THE STORED PC.
                        0060       ;
                        0060       ; OUTPUTS:
                        0060       ;
                        0060       ;         THE ERROR MESSAGE IS DISPLAYED.
                        0060       ;         THE FAO ARGUMENTS ARE POPPED FROM THE STACK.
                        0060       ;         R1,R2 ARE DESTROYED
                        0060       ;-
                        0060       DCL$FORMMSG::                                 ;OUTPUT MESSAGE
        52    8E    D0  0060             MOVL    (SP)+,R2                        ;GET SAVED PC
              51    DD  0063             PUSHL   R1                              ;PUSH FAO COUNT
              50    DD  0065             PUSHL   R0                              ;SET STATUS CODE
  7E    51    02    C1  0067             ADDL3   #2,R1,-(SP)                     ;SET # ARGS IN PUTMSG VECTOR
        50    5E    D0  006B             MOVL    SP,R0                           ;SET ADDRESS OF THE ARGUMENT VECTOR
              0E    10  006E             BSBB    DCL$PUTMSG                      ;SIGNAL PUTMSG VECTOR AT (R0)
        5E    08    C0  0070             ADDL    #8,SP                           ;RESTORE THE STACK
              51  8ED0  0073             POPL    R1                              ;
     5E    6E41     DE  0076             MOVAL   (SP)[R1],SP                     ;
        7E    52    D0  007A             MOVL    R2,-(SP)                        ;RESTORE THE PC
                    05  007D             RSB                                     ;
```

```
                               007E               .SBTTL  PUTMSG OUTPUT ROUTINE
                               007E      ;---
                               007E      ;
                               007E      ;        THIS ROUTINE OUTPUTS A GIVEN PUTMSG MESSAGE VECTOR
                               007E      ;
                               007E      ; INPUTS:
                               007E      ;
                               007E      ;        R10 = ADDRESS OF COMMAND WORK AREA
                               007E      ;        R11 = ADDRESS OF PROCESS WORK AREA
                               007E      ;        R0 = ADDRESS OF PUTMSG VECTOR
                               007E      ;
                               007E      ; OUTPUTS:
                               007E      ;
                               007E      ;        NONE
                               007E      ;---
                               007E      DCL$PUTMSG::
                               007E               DISABLE                            ; DISABLE CONTROL Y/C AST'S
         204C4344 8F      DD   0084               PUSHL   #^A'DCL '                  ; FACILITY NAME
                  5E      DD   008A               PUSHL   SP                         ; MAKE DESCRIPTOR OF NAME
                  03      DD   008C               PUSHL   #3
                  5E      DD   008E               PUSHL   SP                         ; SET ADDRESS OF FACNAM
      04 A0    0C 10      ED   0090               CMPZV   #STS$V_FAC_NO,#STS$S_FAC_NO,4(R0),-
                  03           0095                       #<CLI$_ABKEYW&STS$M_FAC_NO>@-STS$V_FAC_NO
                  02      13   0096               BEQL    10$                        ; BRANCH IF OUR FACILITY
                  6E      D4   0098               CLRL    (SP)                       ; IF NOT OUR FACILITY, ZERO FACNAM PARAMETER
                  00      DD   009A      10$:     PUSHL   #0                         ; NO ACTION ROUTINE
                  50      DD   009C               PUSHL   R0                         ; R0 = ADDRESS OF MESSAGE VECTOR
      00000000'GF 06      FB   009E               CALLS   #6,G^SYS$PUTMSG            ; WRITE THE MESSAGE TO SYS$ERROR,OUTPUT
                               00A5               ENABLE                             ; RE-ENABLE CONTROL/Y AST'S
         03 F0 AA 01      E1   00A7               BBC     #WRK_V_COMMAND,WRK_W_FLAGS(R10),90$ ; BRANCH IF NO IMAGE ACTIVE
               FF51'      30   00AC               BSBW    DCL$CHECK_AST              ; CHECK FOR PENDING AST
                  05           00AF      90$:     RSB
```

```
                    00B0         .SBTTL  OUTPUT FILE MESSAGE OUTPUT (FOR SPAWN)
                    00B0  ;+
                    00B0  ; DCL$SPAWNOUT - OUTPUT FILE MESSAGE OUTPUT
                    00B0  ;
                    00B0  ; THSI ROUTINE IS CALLED TO OUTPUT A MESSAGE TO THE OUTPUT FILE.
                    00B0  ;
                    00B0  ; INPUTS:
                    00B0  ;
                    00B0  ;       R1 = LENGTH OF MESSAGE.
                    00B0  ;       R2 = ADDRESS OF MESSAGE.
                    00B0  ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                    00B0  ;
                    00B0  ; OUTPUTS:
                    00B0  ;
                    00B0  ;       THE MESSAGE IS WRITTEN TO THE OUTPUT FILE AND CONTROL IS RETURNED
                    00B0  ;       TO THE CALLER.
                    00B0  ;
                    00B0  ;       REGISTERS R3, R4, AND R5 ARE PRESERVED ACROSS CALL.
                    00B0  ;-
                    00B0
                    00B0  DCL$SPAWNOUT::                                 ;MESSAGE OUTPUT
   50    18 AB  D0  00B0          MOVL    PRC_L_INDOUTRAB(R11),R0 ;GET ADDRESS OF INDIRECT OUTPUT RAB
04 68 AB    0B  E1  00B4          BBC     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),20$ ;BR IF NOT IN CONTROL Y
   50    0C AB  D0  00B9          MOVL    PRC_L_OUTRAB(R11),R0    ;SET ADDRESS OF OUTPUT FILE RAB
                    00BD  20$:    DISABLE                         ;DISABLE CONTROL Y/C AST'S
   22 A0    51  B0  00C3          MOVW    R1,RAB$W_RSZ(R0)        ;SET SIZE OF OUTPUT RECORD
   28 A0    52  D0  00C7          MOVL    R2,RAB$L_RBF(R0)        ;SET ADDRESS OF OUTPUT RECORD
                    00CB          $PUT    RAB=(R0)                ;OUTPUT RECORD
                    00D4          ENABLE                          ;ENABLE CONTROL Y/C AST'S
                05  00D6          RSB                             ;
```

```
                         00D7          .SBTTL  OUTPUT FILE MESSAGE OUTPUT
                         00D7      ;+
                         00D7      ; DCL$MSGOUT - OUTPUT FILE MESSAGE OUTPUT
                         00D7      ;
                         00D7      ; THSI ROUTINE IS CALLED TO OUTPUT A MESSAGE TO THE OUTPUT FILE.
                         00D7      ;
                         00D7      ; INPUTS:
                         00D7      ;
                         00D7      ;       R1 = LENGTH OF MESSAGE.
                         00D7      ;       R2 = ADDRESS OF MESSAGE.
                         00D7      ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                         00D7      ;
                         00D7      ; OUTPUTS:
                         00D7      ;
                         00D7      ;       THE MESSAGE IS WRITTEN TO THE OUTPUT FILE AND CONTROL IS RETURNED
                         00D7      ;       TO THE CALLER.
                         00D7      ;
                         00D7      ;       REGISTERS R3, R4, AND R5 ARE PRESERVED ACROSS CALL.
                         00D7      ;-
                         00D7
                         00D7      DCL$MSGOUT::                              ;MESSAGE OUTPUT
      50    18 AB   D0   00D7              MOVL    PRC_L_INDOUTRAB(R11),R0 ;GET ADDRESS OF INDIRECT OUTPUT RAB
   04 68 AB    0B   E1   00DB              BBC     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),20$ ;BR IF NOT IN CONTROL Y
      50    0C AB   D0   00E0              MOVL    PRC_L_OUTRAB(R11),R0    ;SET ADDRESS OF OUTPUT FILE RAB
                         00E4      20$:    DISABLE                         ;DISABLE CONTROL Y/C AST'S
      22 A0    51   B0   00EA              MOVW    R1,RAB$W_RSZ(R0)        ;SET SIZE OF OUTPUT RECORD
      28 A0    52   D0   00EE              MOVL    R2,RAB$L_RBF(R0)        ;SET ADDRESS OF OUTPUT RECORD
                         00F2              $PUT    RAB=(R0)                ;OUTPUT RECORD
                         00FB              ENABLE                          ;ENABLE CONTROL Y/C AST'S
   03 F0 AA    01   E1   00FD              BBC     #WRK_V_COMMAND,WRK_W_FLAGS(R10),30$ ;IF CLR, NO COMMAND EXECUTION
            FEFB'   30   0102              BSBW    DCL$CHECK_AST           ;CHECK FOR PENDING AST
                    05   0105      30$:    RSB                             ;
```

```
                          0106   304
                          0106   305             .SBTTL  SET FLUSH RATE COMMAND
                          0106   306 ;+
                          0106   307 ; DCL$SETFLUSH - FLUSH OUTPUT FILE
                          0106   308 ;
                          0106   309 ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO SET THE OUTPUT BUFFER
                          0106   310 ; FLUSH RATE.
                          0106   311 ;
                          0106   312 ; INPUTS:
                          0106   313 ;
                          0106   314 ;           R10 = BASE ADDRESS OF COMMAND WORK AREA.
                          0106   315 ;           R11 = BASE ADDRESS OF PROCESS WORK AREA.
                          0106   316 ;
                          0106   317 ; OUTPUTS:
                          0106   318 ;
                          0106   319 ;           THE FLUSH RATE IS SET.
                          0106   320 ;-
                          0106   321 DCL$SETFLUSH::                                          ;
                          0106   322         STATUS  NORMAL                                 ; ASSUME SUCCESS
                06    E1  010D   323         BBC     #PRC_V_MODE,-                          ; IGNORE IF INTERACTIVE PROCESS
          30 68 AB        010F   324                 PRC_Q_FLAGS(R11),30$                   ;
             FEEB'    30  0112   325         BSBW    DCL$GETDVAL                            ; SKIP PAST OPTION KEYWORD
             FEE8'    30  0115   326         BSBW    DCL$GETDVAL                            ; WAS A VALUE SPECIFIED?
          04    55    D1  0118   327         CMPL    R5,#PTR_K_ENDLINE                      ; NO, THEN SIMPLY FORCE A FLUSH NOW
                1E    13  011B   328         BEQL    20$                                    ;
          7E    51    7D  011D   329         MOVQ    R1,-(SP)                               ; BUILD DESCRIPTOR FOR DELTA TIME ST
          53    5E    D0  0120   330         MOVL    SP,R3                                  ; SAVE ADDRESS OF QUAD WORD DECRIPTO
                          0123   331         $BINTIM_S TIMBUF=(R3),-                        ; CONVERT TIME TO BINARY DELTA FORMA
                          0123   332                 TIMADR=(R3)
             11 50    E9  012E   333         BLBC    R0,30$                                 ; BR IF ERROR IN TIME
       00D0 CB  8E    7D  0131   334         MOVQ    (SP)+,PRC_Q_FLUSHTIME(R11)             ; SAVE THE FLUSH TIME INTERVAL
                0B    10  0136   335         BSBB    DCL$SET_TIMER                          ; SET THE TIMER
             07 50    E9  0138   336         BLBC    R0,30$                                 ; BRANCH IF ERROR
   00000173'EF  00    FB  013B   337 20$:    CALLS   #0,FLUSH                               ; FLUSH THE OUTPUT BUFFER
                      05  0142   338 30$:    RSB
```

```
        0143          .SBTTL  SET THE FLUSH TIMER
        0143    ;+
        0143    ; DCL$SET_TIMER - SET THE FLUSH TIMER
        0143    ;
        0143    ; THIS ROUTINE SETS THE OUTPUT BUFFER FLUSH TIMER.
        0143    ;
        0143    ; INPUTS:
        0143    ;
        0143    ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
        0143    ;
        0143    ; OUTPUTS:
        0143    ;
        0143    ;       R0 = $SETIMR STATUS
        0143    ;       THE TIMER IS SET.
        0143    ;-
        0143
        0143    DCL$SET_TIMER::
        0143            $CANTIM_S REQIDT=R11                    ;CANCEL PREVIOUS FLUSH TIMERS
        014E            $SETIMR_S EFN=#EXE$C_SYSEFN,-           ;SET TIMER FOR SPECIF ED TIME
        014E                    DAYTIM=PRC_Q_FLUSHTIME(R11),-   ;
        014E                    ASTADR=FLUSH_AST,-              ;
        014E                    REQIDT=R11                      ;
  05    0167            RSB                                     ;
```

```
                              0168          .SBTTL  FLUSH OUTPUT FILE
                              0168  ;+
                              0168  ; FLUSH_AST - FLUSH OUTPUT FILE
                              0168  ; FLUSH - FLUSH OUTPUT FILE
                              0168  ;
                              0168  ; THIS ROUTINE IS CALLED TO FLUSH THE OUTPUT FILE.
                              0168  ;
                              0168  ; INPUTS:
                              0168  ;
                              0168  ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                              0168  ;
                              0168  ; OUTPUTS:
                              0168  ;
                              0168  ;       R0 = STATUS
                              0168  ;-
                              0168          .ENABLE LSB
                              0168
                              0168  FLUSH_AST:                                                  ;FLUSH OUTPUT
                        087C  0168          .WORD   ^M<R2,R3,R4,R5,R6,R11>                      ;
              5B  04 AC   D0  016A          MOVL    4(AP),R11                                   ;GET ADDRESS OF PRC DATA STRUCTURE
                56   01   D0  016E          MOVL    #1,R6                                       ;RESET THE TIMER
                     04   11  0171          BRB     10$                                         ;
                              0173
                        087C  0173  FLUSH:  .WORD   ^M<R2,R3,R4,R5,R6,R11>                          ;FLUSH OUTPUT
                     56   D4  0175          CLRL    R6                                          ;DO NOT RESET THE TIMER
                              0177  ;
                              0177  ; INITIALIZE TEMPORARY RAB
                              0177  ;
              5E   BC AE  9E  0177  10$:    MOVAB   -RAB$C_BLN(SP),SP                           ;ALLOCATE A RAB ON STACK
6E   0044 8F  00   6E  00 2C  017B          MOVC5   #0,(SP),#0,#RAB$C_BLN,(SP)                  ;ZERO THE RAB(R1 NOW CONTAINS
                              0183                                                              ;       ADDRESS OF RAB
                              0183          ASSUME  RAB$B_BID EQ 0
                              0183          ASSUME  RAB$B_BLN EQ RAB$B_BID+1
                              0183
          61   4401 8F    B0  0183          MOVW    #<RAB$C_BID+<RAB$C_BLN@8>>,(R1)  ;FILL IN BID AND BLN
                              0188
              50   18 AB  D0  0188          MOVL    PRC_L_INDOUTRAB(R11),R0                     ;GET ADDRESS OF INDIRECT OUTPUT RAB
          04 68 AB    0B  E1  018C          BBC     #PRC_V_YLEVEL,PRC_W_FLAGS(R11),20$ ;BR IF NOT IN CONTROL Y
              50   0C AB  D0  0191          MOVL    PRC_L_OUTRAB(R11),R0                        ;SET ADDRESS OF OUTPUT FILE RAB
                   02 A0  B0  0195  20$:    MOVW    RAB$W_ISI(R0),-                             ;COPY PPF ISI TO TEMP. RAB
                   02 A1      0198                  RAB$W_ISI(R1)
                              019A          $FLUSH  RAB=(R1)                                    ;OUTPUT RECORD
                   03 56  E9  01A3          BLBC    R6,30$                                      ;BRANCH IF NOT RESETING TIMER
                    FF9A  30  01A6          BSBW    DCL$SET_TIMER                               ;RESET THE TIMER
                50   01   D0  01A9  30$:    MOVL    #1,R0                                       ;SET SUCCESSFUL STATUS
                          04  01AC          RET                                                 ;
                              01AD
                              01AD          .DISABLE LSB
                              01AD
                              01AD          .END
```

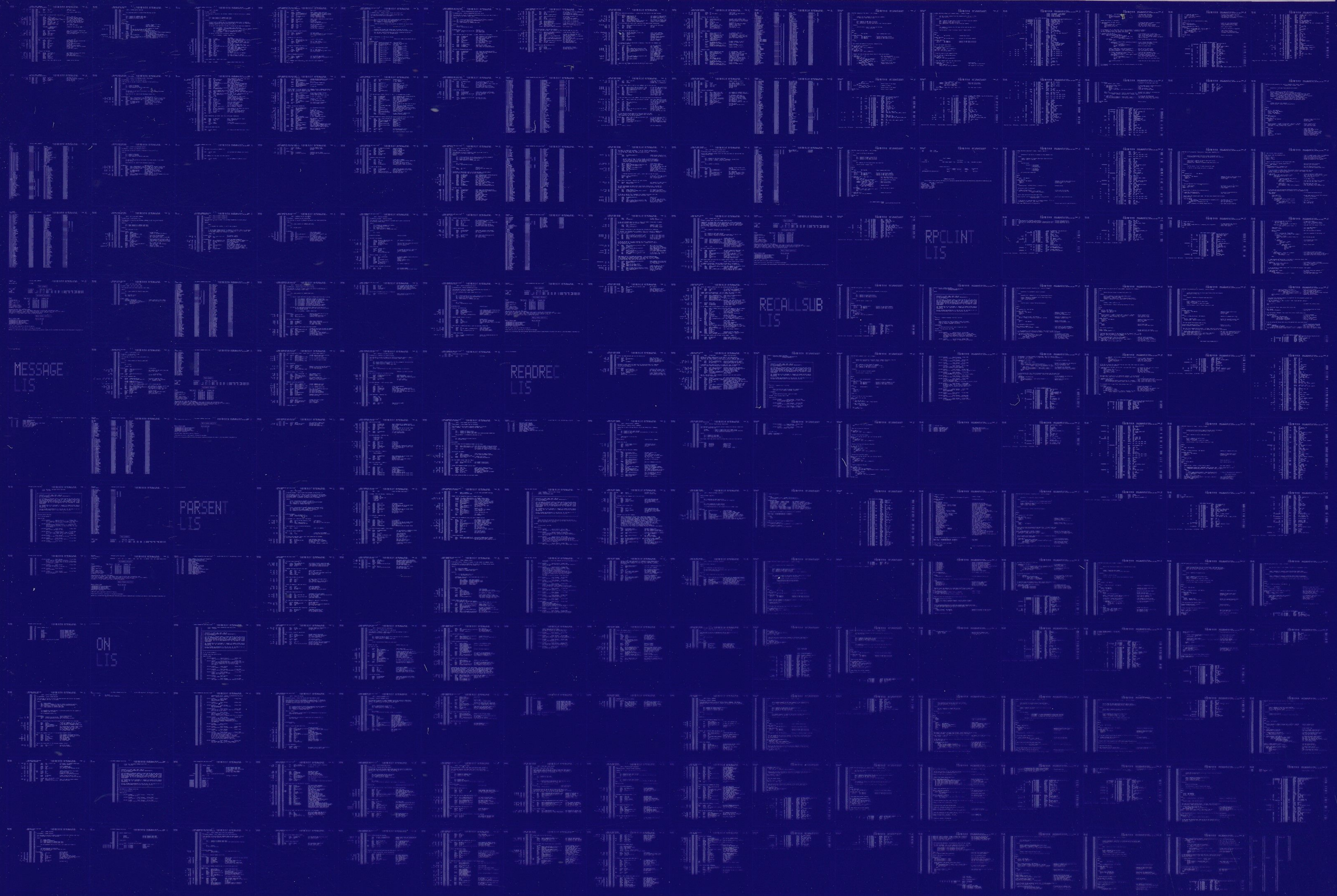
```
$$.TMP1           = 00000001            PRC_L_RESTART       00000058
$$.TMP2           = 00000061            PRC_L_SAVAP         00000000
$$T1              = 00000000            PRC_L_SAVFP         00000004
CLI$_ABKEYW       = 00038010            PRC_L_SEVERITY      00000050
CLI$_CMDSEG       = 00038248            PRC_L_SPWN          000000C0
CLI$_NORMAL       = 00030001            PRC_L_STACKLM       000000A4
DCL$CHECK_AST       ********   X   02   PRC_L_STACKPT       000000A0
DCL$DISABLE         ********   X   02   PRC_L_STATUS        00000054
DCL$ERRORMSG        00000000 RG    02   PRC_L_STS           00000084
DCL$FORMMSG         00000060 RG    02   PRC_L_STV           00000088
DCL$GETDVAL         ********   X   02   PRC_L_SYMBOL        00000060
DCL$MARKEDTOKEN     ********   X   02   PRC_L_TMBX          00000074
DCL$MSGOUT          000000D7 RG    02   PRC_L_TRMLIST       00000010
DCL$PUTMSG          0000007E RG    02   PRC_Q_ALLOCREG      00000020
DCL$SETFLUSH        00000106 RG    02   PRC_Q_COMMAND       000000E0
DCL$SET_TIMER       00000143 RG    02   PRC_Q_FLUSHTIME     000000D0
DCL$SPAWNOUT        000000B0 RG    02   PRC_Q_GLOBAL        00000028
EXE$C_SYSEFN        ********   X   02   PRC_Q_IMAGENAME     000000D8
FLUSH               00000173 R     02   PRC_Q_KEYPAD        00000040
FLUSH_AST           00000168 R     02   PRC_Q_LABEL         00000030
PRC_B_CONTINUE      000000F3            PRC_Q_LOCAL         00000038
PRC_B_DEFRADIX      000000AE            PRC_Q_SAVEPRIV      000000E8
PRC_B_EXMDEPMOD     000000AD            PRC_T_OUTDVI        0000011C
PRC_B_EXMDEPWID     000000AC            PRC_V_MODE        = 00000006
PRC_B_EXONLYL       0000012D            PRC_V_YLEVEL      = 0000000B
PRC_B_FLAGS2        000000AF            PRC_W_ASTIOSB       000000C6
PRC_B_IMGFLAG       00000078            PRC_W_ASTRETN       000000C8
PRC_B_OUTFLAGS      0000012C            PRC_W_ASTSTATUS     000000C4
PRC_B_PROMPTLEN     000000F0            PRC_W_ATTMBX        0000007A
PRC_C_LENGTH        00000534            PRC_W_FLAGS         00000068
PRC_G_COMMANDS      00000133            PRC_W_INPCHAN       00000064
PRC_G_PROMPT        000000F4            PRC_W_ONLEVEL       0000006A
PRC_K_LENGTH        00000534            PRC_W_OUTIFI        00000114
PRC_L_CURRKEY       00000048            PRC_W_OUTISI        00000116
PRC_L_EXMDEPADR     000000A8            PRC_W_OUTMBXCHN     000000CA
PRC_L_EXTARG        00000094            PRC_W_OUTMBXREF     000000CE
PRC_L_EXTBLK        0000008C            PRC_W_OUTMBXSIZ     000000CC
PRC_L_EXTCOD        0000009C            PRC_W_PMPTCTRL      000000F1
PRC_L_EXTHND        00000090            PRC_W_WAITIOSB      00000066
PRC_L_EXTPRM        00000098            PTR_B_LEVEL         00000004
PRC_L_IDFLNK        000000BC            PTR_B_NUMBER        00000005
PRC_L_IMGACTSTS     00000080            PTR_B_PARMCNT       00000006
PRC_L_INDCLOCK      0000007C            PTR_B_VALUE         00000000
PRC_L_INDEPTH       0000005C            PTR_C_LENGTH        0000000C
PRC_L_INDFAB        0000001C            PTR_K_ENDLINE     = 00000004
PRC_L_INDINPRAB     00000014            PTR_K_LENGTH        0000000C
PRC_L_INDOUTRAB     00000018            PTR_L_DESCR         00000000
PRC_L_INPRAB        00000008            PTR_L_ENTITY        00000008
PRC_L_LASTKEY       0000004C            RAB$B_BID         = 00000000
PRC_L_LSTSTATUS     000000B0            RAB$B_BLN         = 00000001
PRC_L_ONCTLY        000000B8            RAB$C_BID         = 00000001
PRC_L_ONERROR       0000006C            RAB$C_BLN         = 00000044
PRC_L_OUTOFBAND     000000B4            RAB$L_RBF         = 00000028
PRC_L_OUTRAB        0000000C            RAB$W_ISI         = 00000002
PRC_L_OUTRABCTX     00000118            RAB$W_RSZ         = 00000022
PRC_L_PPFLIST       00000070            STS$M_FAC_NO      = 0FFF0000
PRC_L_RECALLPTR     0000012F            STS$S_FAC_NO      = 0000000C
```

```
STS$V_FAC_NO                     = 00000010
STS$V_INHIB_MSG                  = 0000001C
SYS$BINTIM                         ********   GX    02
SYS$CANTIM                         ********   GX    02
SYS$FLUSH                          ********   GX    02
SYS$PUT                            ********   GX    02
SYS$PUTMSG                         ********    X    02
SYS$SETIMR                         ********   GX    02
WRK_B_CMDOPT                       FFFFFFC3
WRK_B_MAXPARM                      FFFFFFD0
WRK_B_MINPARM                      FFFFFFD1
WRK_B_PARMCNT                      FFFFFFCE
WRK_B_PARMSUM                      FFFFFFCF
WRK_B_RECALLCNT                    FFFFFFC5
WRK_B_VALLEV                       FFFFFFC4
WRK_B_VERBTYP                      FFFFFFC2
WRK_C_LENGTH                       FFFFF486
WRK_G_BUFFER                       FFFFF492
WRK_G_INPBUF                       FFFFF896
WRK_G_RESULT                       FFFFF9B6
WRK_K_LENGTH                       FFFFF486
WRK_L_CHARPTR                      FFFFF48E
WRK_L_DISALLOW                     FFFFFFE6
WRK_L_ERRORRTN                     FFFFF9AE
WRK_L_EXPANDPTR                    FFFFF486
WRK_L_IMAGE                        FFFFFFE2
WRK_L_MARKPTR                      FFFFF48A
WRK_L_PAROUT                       FFFFFFD2
WRK_L_PMPTADDR                     FFFFF9A2
WRK_L_PROMPTRTN                    FFFFF9A6
WRK_L_PROPTR                       FFFFFFC6
WRK_L_QUABLK                       FFFFFFCA
WRK_L_READRTN                      FFFFF9AA
WRK_L_RECALLPTR                    FFFFFFEA
WRK_L_RSLEND                       FFFFFFB6
WRK_L_RSLNXT                       FFFFFFBA
WRK_L_SAVAP                        FFFFFFF8
WRK_L_SAVFP                        FFFFFFFC
WRK_L_SAVSP                        FFFFFFF4
WRK_L_SIGNALRTN                    FFFFFFD6
WRK_L_SPECRTN                      FFFFF9B2
WRK_L_TAB_VEC                      FFFFFFDE
WRK_L_VERB                         FFFFFFBE
WRK_V_COMMAND                    = 00000001
WRK_W_FLAGS                        FFFFFFF0
WRK_W_FLAGS2                       FFFFFFF2
WRK_W_IMGCHAN                      FFFFFFEE
WRK_W_PMPTLEN                      FFFFF99E


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

PSECT name                        Allocation          PSECT No.  Attributes
----------                        ----------          ---------  ----------
.  ABS  .                         00000000  (     0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                             FFFFFFFC  (     0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
```

```
DCL$ZCODE                        000001AD  (  429.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC BYTE

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                      9    00:00:00.08    00:00:00.35
Command processing                 82    00:00:00.74    00:00:04.24
Pass 1                            287    00:00:11.46    00:00:33.03
Symbol table sort                   0    00:00:01.52    00:00:04.19
Pass 2                             71    00:00:02.01    00:00:04.89
Symbol table output                19    00:00:00.18    00:00:00.81
Psect synopsis output               2    00:00:00.02    00:00:00.02
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              470    00:00:16.01    00:00:47.53

The working set limit was 1350 pages.
57392 bytes (113 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1035 non-local and 15 local symbols.
413 source lines were read in Pass 1, producing 14 object records in Pass 2.
48 pages of virtual memory were used to define 33 macros.

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

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[SYSLIB]SYSBLDMLB.MLB;1                       0
_$255$DUA28:[DCL.OBJ]DCL.MLB;1                            8
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                            0
_$255$DUA28:[SYSLIB]STARLET.MLB;2                        19
TOTALS (all libraries)                                   27

1280 GETS were required to define 27 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:MESSAGE/OBJ=OBJ$:MESSAGE MSRC$:MESSAGE/UPDATE=(ENH$:MESSAGE)+EXECML$/LIB+LIB$:DCL/LIB+SYS$LIBRARY:SYSBLDMLB/LIB
```

0072 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



RPCLINT
LIS

RECALLSUB
LIS

MESSAGE
LIS

READREC
LIS

PARSENT
LIS

ON
LIS