```
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
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
DDDDDDDDDDDD        CCCCCCCCCCCC  LLLLLLLLLLLLLLL
```

FILEID**IMAGECTRL

```
IIIIII    MM      MM    AAAAAA      GGGGGGGG  EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL
IIIIII    MM      MM    AAAAAA      GGGGGGGG  EEEEEEEEEE    CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL
  II      MMMM  MMMM   AA      AA   GG        EE          CC              TT      RR      RR  LL
  II      MMMM  MMMM   AA      AA   GG        EE          CC              TT      RR      RR  LL
  II      MM  MM  MM   AA      AA   GG        EE          CC              TT      RR      RR  LL
  II      MM  MM  MM   AA      AA   GG        EE          CC              TT      RR      RR  LL
  II      MM      MM   AA      AA   GG        EEEEEEEE    CC              TT      RRRRRRRR    LL
  II      MM      MM   AA      AA   GG        EEEEEEEE    CC              TT      RRRRRRRR    LL
  II      MM      MM   AAAAAAAAAA   GG  GGGGGG  EE          CC              TT      RR  RR      LL
  II      MM      MM   AAAAAAAAAA   GG  GGGGGG  EE          CC              TT      RR  RR      LL
  II      MM      MM   AA      AA   GG      GG  EE          CC              TT      RR    RR    LL                  ....
  II      MM      MM   AA      AA   GG      GG  EE          CC              TT      RR    RR    LL                  ....
IIIIII    MM      MM   AA      AA     GGGGGG    EEEEEEEEEE    CCCCCCCC        TT      RR      RR  LLLLLLLLLL      ....
IIIIII    MM      MM   AA      AA     GGGGGG    EEEEEEEEEE    CCCCCCCC        TT      RR      RR  LLLLLLLLLL      ....

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  IMAGECTRL - IMAGE CONTROL
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  ; IMAGE CONTROL DCLS COMMAND EXECUTION
0000    29  ;
0000    30  ;       CONTINUE IMAGE EXECUTION
0000    31  ;       DEBUG IMAGE EXECUTION
0000    32  ;       STOP IMAGE EXECUTION
0000    33  ;
0000    34  ; D. N. CUTLER 4-APR-77
0000    35  ;
0000    36  ; MODIFIED BY:
0000    37  ;
0000    38  ;       V03-006 HWS0071         Harold Schultz          04-Jun-1984
0000    39  ;               When finished with skipping data records in the input
0000    40  ;               stream of an image being run down, set EOL in the input
0000    41  ;               buffer following the last record read.
0000    42  ;
0000    43  ;       V03-005 HWS0036         Harold Schultz          21-Mar-1984
0000    44  ;               Use PRC_V_IRUNDWN to indicate whether or not an image
0000    45  ;               has been run down by DCL
0000    46  ;
0000    47  ;       V03-004 HWS0026         Harold Schultz          09-Mar-1984
0000    48  ;               When shutting down an image, check if device is a
0000    49  ;               record-oriented device rather than a terminal.
0000    50  ;
0000    51  ;       V03-003 PCG0005         Peter George            15-Jun-1983
0000    52  ;               Create DCL$RMSRUNDWN.
0000    53  ;
0000    54  ;       V03-002 PCG0004         Peter George            24-Feb-1983
0000    55  ;               Remove SETBIT WRK_V_NOSTAT from CONTINUE and STOP.
0000    56  ;
0000    57  ;       V03-001 PCG0003         Peter George            21-Jan-1983
```

```
0000     58 ;
0000     59 ;  Remove code that is duplicated in DCL$LOGOUT
0000     60 ;---  from DCL$STOP.
```

```
                                0000     62 ;
                                0000     63 ; MACRO LIBRARY CALLS
                                0000     64 ;
                                0000     65
                                0000     66          $PPDDEF                         ;PROCESS PERMANENT DATA AREA
                                0000     67          PRCDEF                          ;DEFINE PROCESS WORK AREA
                                0000     68          WRKDEF                          ;DEFINE COMMAND WORK AREA
                                0000     69          PTRDEF                          ;DEFINE RESULT PARSE DESCRIPTOR FORMAT
                                0000     70          $DEVDEF                         ;DEFINE DEVICE CHARACTERISTIC BITS
                                0000     71          $PSLDEF                         ;DEFINE PROCESSOR STATUS FIELDS
                                0000     72          $RABDEF                         ;DEFINE RAB OFFSETS
                                0000     73          $SSDEF                          ;DEFINE SYSTEM STATUS VALUES
                                0000     74          $CLIMSGDEF                      ;DEFINE ERROR/STATUS VALUES
                                0000     75
                                0000     76 ;
                                0000     77 ; LOCAL DATA
                                0000     78 ;
                                0000     79 ; HEX CONVERSION TABLE
                                0000     80 ;
                                0000     81
                           00000000     82          .PSECT  DCL$ZCODE,BYTE,RD,NOWRT
34 35 36 37 38 39 41 42 43 44 45 46  0000     83 HEXTAB: .ASCII  /FEDCBA9876543210/       ;
                      30 31 32 33  000C
```

```
                     0010    85          .SBTTL  CONTINUE IMAGE EXECUTION
                     0010    86 ;+
                     0010    87 ; DCL$CONTINUE - CONTINUE IMAGE EXECUTION
                     0010    88 ;
                     0010    89 ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE CONTINUE DCLS
                     0010    90 ; COMMAND.
                     0010    91 ;
                     0010    92 ; INPUTS:
                     0010    93 ;
                     0010    94 ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                     0010    95 ;       R9 = ADDRESS OF SCRATCH STACK.
                     0010    96 ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                     0010    97 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                     0010    98 ;
                     0010    99 ; OUTPUTS:
                     0010   100 ;
                     0010   101 ;       IF A PREVIOUS IMAGE WAS INTERRUPTED VIA A CONTROL Y AST, THEN THE
                     0010   102 ;       CURRENT COMMAND CONTEXT IS REMOVED FROM THE STACK AND CONTROL IS
                     0010   103 ;       RETURNED TO THE IMAGE. OTHERWISE THIS COMMAND IS A NOPERATION.
                     0010   104 ;-
                     0010   105
                     0010   106          .ENABL  LSB
                     0010   107 DCL$CONTINUE::                                ;CONTINUE IMAGE EXECUTION
          00B9   30  0010   108          BSBW    TESTMODE                     ;TEST PREVIOUS MODE
03 00AF CB   04   E1  0013   109          BBC     #PRC_V_PRIV,PRC_B_FLAGS2(R11),10$  ;BR IF UNPRIVILEGED IMAGE
          00DE   30  0019   110          BSBW    RESTORE_PRIVS                ;RESTORE IMAGE PRIVILEGE
                04   001C   111 10$:     RET                                  ;
```

```
                        001D   113          .SBTTL  DEBUG IMAGE EXECUTION
                        001D   114 ;+
                        001D   115 ; DCL$DEBUG - DEBUG IMAGE EXECUTION
                        001D   116 ;
                        001D   117 ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE DEBUG DCLS
                        001D   118 ; COMMAND.
                        001D   119 ;
                        001D   120 ; INPUTS:
                        001D   121 ;
                        001D   122 ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                        001D   123 ;       R9 = ADDRESS OF SCRATCH STACK.
                        001D   124 ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        001D   125 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        001D   126 ;
                        001D   127 ; OUTPUTS:
                        001D   128 ;
                        001D   129 ;       IF A PREVIOUS IMAGE WAS INTERRUPTED VIA A CONTROL Y AST, THEN A
                        001D   130 ;       DEBUG EXCEPTION IS GENERATED FOR THE IMAGE. OTHERWISE THIS COMMAND
                        001D   131 ;       IS A NOPERATION.
                        001D   132 ;-
                        001D   133
                        001D   134 DCL$DEBUG::                                  ;DEBUG IMAGE EXECUTION
              00AC  30  001D   135          BSBW    TESTMODE                  ;TEST PREVIOUS MODE
10 AD   2D'AF   9E      0020   136          MOVAB   B^20$,16(FP)              ;RESET AST RETURN ADDRESS
 7E   0B   16   78      0025   137          ASHL    #PSL$V_PRVMOD,#<PSL$C_SUPER@2>! - ;CONSTRUCT PROPER PSL
                        0029   138                  PSL$C_USER,-(SP)          ;
        F0 AF   9F      0029   139          PUSHAB  10$                       ;SET PC
                02      002C   140          REI                               ;
                        002D   141
                        002D   142 ;
                        002D   143 ; CONTROL IS REGAINED AT THIS POINT WITH:
                        002D   144 ;
                        002D   145 ;       00(SP) = NUMBER OF AST ARGUMENTS (ALWAYS 5).
                        002D   146 ;       04(SP) = AST PARAMETER.
                        002D   147 ;       08(SP) = SAVED R0.
                        002D   148 ;       12(SP) = SAVED R1.
                        002D   149 ;       16(SP) = IMAGE PC.
                        002D   150 ;       20(SP) = IMAGE PSL.
                        002D   151 ;
                        002D   152
     5E   08   C0       002D   153 20$:     ADDL    #8,SP                     ;REMOVE NUMBER OF ARGUMENTS AND PARAMETER
          03   BA       0030   154          POPR    #^M<R0,R1>                ;RESTORE SAVED REGISTERS
7E   046C 8F   3C       0032   155          MOVZWL  #SS$_DEBUG,-(SP)          ;SET EXCEPTION NAME
          03   DD       0037   156          PUSHL   #3                        ;SET NUMBER OF EXCEPTION ARGUMENTS
 00000000'9F   17       0039   157          JMP     @#EXE$REFLECT             ;REFLECT EXCEPTION
                        003F   158          .DSABL  LSB
```

```
                     003F   160          .SBTTL  STOP IMAGE EXECUTION
                     003F   161 ;+
                     003F   162 ; DCL$STOP - STOP IMAGE EXECUTION
                     003F   163 ;
                     003F   164 ; THIS ROUTINE IS CALLED AS AN INTERNAL COMMAND TO EXECUTE THE STOP DCLS
                     003F   165 ; COMMAND.
                     003F   166 ;
                     003F   167 ; INPUTS:
                     003F   168 ;
                     003F   169 ;       R8 = ADDRESS OF SCRATCH BUFFER DESCRIPTOR.
                     003F   170 ;       R9 = ADDRESS OF SCRATCH STACK.
                     003F   171 ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                     003F   172 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                     003F   173 ;
                     003F   174 ; OUTPUTS:
                     003F   175 ;
                     003F   176 ;       IF A PROCESS NAME OR IDENTIFICATION IS SPECIFIED, THEN THAT PROCESS IS
                     003F   177 ;       DELETED.
                     003F   178 ;
                     003F   179 ;       IF THE JOB IS A NONINTERACTIVE JOB, THEN THE JOB IS LOGGED OFF THE SYSTEM
                     003F   180 ;       WITH A STATUS OF NORMAL COMPLETION. OTHERWISE ALL INDIRECT FILE LEVELS ARE
                     003F   181 ;       UNSTACKED AND A TEST IS MADE TO DETERMINE IF AN IMAGE WAS INTERRUPTED VIA
                     003F   182 ;       A CONTROL C/Y. IF A PREVIOUS IMAGE WAS INTERRUPTED, THEN THE CONTEXT OF THE
                     003F   183 ;       RUN COMMAND THAT INITIATED IMAGE EXECUTION IS REMOVED FROM THE STACK AND
                     003F   184 ;       RMS-32 IS CALLED TO CLOSE ALL OPEN IMAGE FILES. OTHERWISE NO OPERATION IS
                     003F   185 ;       PERFORMED.
                     003F   186 ;-
                     003F   187
                     003F   188 DCL$STOP::                                   ;STOP IMAGE EXECUTION
        FFBE'   30   003F   189          BSBW    DCL$GETDVAL                ;GET DESCRIPTOR VALUES
     55   04   D1   0042   190          CMPL    #PTR_K_ENDLINE,R5          ;END OF LINE?
          3E   12   0045   191          BNEQ    40$                        ;IF NEQ NO
       5C AB   D5   0047   192 10$:     TSTL    PRC_L_INDEPTH(R11)         ;INDIRECT LEVEL ZERO?
          05   13   004A   193          BEQL    20$                        ;IF EQL YES
        FFB1'   30   004C   194          BSBW    DCL$UNSTACK                ;UNSTACK INDIRECT LEVEL
          F6   11   004F   195          BRB     10$                        ;
          06   E1   0051   196 20$:     BBC     #PRC_V_MODE,-              ;IF SET, NONINTERACTIVE JOB
  03 68 AB        0053   197                  PRC_W_FLAGS(R11),25$       ;
        FFA7'   31   0056   198          BRW     DCL$ABORT                  ;LOG PROCESS OUT
        0070    30   0059   199 25$:     BSBW    TESTMODE                   ;TEST PREVIOUS MODE
 5A   FC AA   D0   005C   200          MOVL    WRK_L_SAVFP(R10),R10       ;RESTORE SAVED WRK ADDRESS
        0151    30   0060   201          BSBW    DCL$SHUTDOWN               ;CLOSE FILES OF PREVIOUS IMAGE
00000000'GF   D4   0063   202          CLRL    G^CTL$GL_CLINTOWN          ;ZERO CLINT OWN STORAGE POINTER
00000000'GF   D4   0069   203          CLRL    G^CTL$GL_DCLPRSOWN         ;ZERO DCL PARSE OWN STORAGE
                   006F   204          $RUNDWN_S #PSL$C_USER              ;RUN DOWN PREVIOUS IMAGE
          18   8A   0078   205          BICB    #<PRC_M_EXEONLY ! PRC_M_PRIV>,- ;SINCE IMAGE IS NOW GONE
    00AF  CB        007A   206                  PRC_B_FLAGS2(R11)          ;NO NEED TO PROTECT IT
                   007D   207          CLRBIT  PRC_V_IRUNDWN,PRC_B_IMGFLAG(R11)  ;INDICATE THAT IMAGE IS RUNDOWN
     5D   5A   D0   0081   208          MOVL    R10,FP                     ;RESET FP SO YLEVEL WRK IS DEALLOCATED
                   0084   209                                             ;ON RETURN TO DCL$RESTART.
                05 0084   210          RSB                                ;
                   0085   211
                   0085   212 ;
                   0085   213 ; DELETE PROCESS
                   0085   214 ;
                   0085   215
          79   D4   0085   216 40$:     CLRL    -(R9)                      ;CLEAR PROCESS IDENTIFICATION
```

```
                57   58   D0  0087   217          MOVL    R8,R7                   ;COPY ADDRESS OF SCRATCH DESCRIPTOR
                55   00   D1  008A   218          CMPL    #PTR_K_COMDQUAL,R5      ;COMMAND QUALIFIER SPECIFIED?
                     26   12  008D   219          BNEQ    70$                     ;IF NEQ NO
                   FF6E'  30  008F   220          BSBW    DCL$GETDVAL             ;GET VALUE PARAMETERS
                52   51   7D  0092   221          MOVQ    R1,R2                   ;SAVE VALUE PARAMETERS
                     52   D7  0095   222  50$:    DECL    R2                      ;ANY MORE CHARACTERS TO CONVERT?
                     12   19  0097   223          BLSS    60$                     ;IF LSS NO
          FF61 CF    10   83  3A  0099   224          LOCC    (R3)+,#16,HEXTAB        ;SEARCH FOR HEX CHARACTER MATCH
                     23   13  009F   225          BEQL    80$                     ;IF EQL VALUE SYNTAX ERROR
                69   10   C4  00A1   226          MULL    #16,(R9)                ;SCALE ACCUMULATED RESULT
                69   50   C0  00A4   227          ADDL    R0,(R9)                 ;ADD IN NEXT DIGIT
                     69   D7  00A7   228          DECL    (R9)                    ;SUBTRACT OUT CHARACTER COUNT
                     EA   11  00A9   229          BRB     50$                     ;
                   FF52'  30  00AB   230  60$:    BSBW    DCL$GETDVAL             ;GET DESCRIPTOR VALUES
                55   04   D1  00AE   231          CMPL    #PTR_K_ENDLINE,R5       ;END OF LINE?
                     02   12  00B1   232          BNEQ    70$                     ;IF NEQ NO
                     57   D4  00B3   233          CLRL    R7                      ;CLEAR DESCRIPTOR ADDRESS
                68   51   7D  00B5   234  70$:    MOVQ    R1,(R8)                 ;SAVE PROCESS NAME PARAMETERS
                          00B8   235          $DELPRC_S (R9),(R7)             ;DELETE PROCESS
                          05  00C3   236          RSB                             ;
                          00C4   237  80$:    STATUS  IVVALU                  ;SET INVALID VALUE SYNTAX STATUS
                          05  00CB   238          RSB                             ;
```

```
                        00CC    240          .SBTTL  TEST PREVIOUS MODE
                        00CC    241 ;
                        00CC    242 ; SUBROUTINE TO TEST PREVIOUS MODE AND DISABLE CONTROL Y AST
                        00CC    243 ;
                        00CC    244
                        00CC    245 TESTMODE:                                   ;TEST PREVIOUS MODE
                        00CC    246          SETBIT  PRC_V_DISABL,PRC_W_FLAGS(R11) ;DISABLE CONTROL Y AST
0A 68 AB    0B  E5      00D0    247          BBCC    #PRC_V_YLEVEL,PRC_W_FLAGS(R11),10$ ;IF CLR, NOT AT CONTROL Y/C LEVEL
05 00 BB    18  E1      00D5    248          BBC     #PSL$V_CURMOD,@PRC_L_SAVAP(R11),10$ ;IF CLR, PREVIOUS MODE SUPERVISO
   6B   F8 AA   7D      00DA    249          MOVQ    WRK_L_SAVAP(R10),PRC_L_SAVAP(R11) ;RESTORE ARGUMENT AND FRAME POINTE
                05      00DE    250          RSB                                     ;
                        00DF    251
                        00DF    252 ;
                        00DF    253 ; PREVIOUS MODE SUPERVISOR
                        00DF    254 ;
                        00DF    255
                8E      D5      00DF    256 10$:     TSTL    (SP)+                   ;REMOVE RETURN FROM STACK
                        00E1    257          STATUS  NORMAL                  ;SET COMPLETION STATUS
                05      00E8    258          RSB                             ;
```

```
                     00E9   260          .SBTTL  SAVE/RESTORE IMAGE PRIVILEGES
                     00E9   261 ;+
                     00E9   262 ; DCL$SAVE_PRIVS - SAVE PRIVILEGED IMAGE PRIVILEGES
                     00E9   263 ;                  SET IMAGE PRIVILEGES TO PROCESS PRIVILEGES
                     00E9   264 ; RESTORE_PRIVS  - RESTORE PRIVILEGED IMAGE PRIVILEGES FROM
                     00E9   265 ;                  SAVED COPY
                     00E9   266 ;
                     00E9   267 ; INPUTS:
                     00E9   268 ;
                     00E9   269 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                     00E9   270 ;       PRC_Q_SAVEPRIV(R11) - SAVED IMAGE PRIVILEGE TO BE USED BY
                     00E9   271 ;             RESTORE_PRIVS
                     00E9   272 ;
                     00E9   273 ; OUTPUTS:
                     00E9   274 ;
                     00E9   275 ;       R0,R1,R2 DESTROYED OTHERS PRESERVED
                     00E9   276 ;       PRC_Q_SAVEPRIV(R11) - PREVIOUS VALUE OF IMAGE PRIVILEGES
                     00E9   277 ;-
                     00E9   278 DCL$SAVE_PRIVS::
                     00E9   279          $SETPRV_S PRMFLG=#1,-   ;READ PROCESS PERMANENT PRIVILEGES
                     00E9   280                    PRVPRV=PRC_Q_SAVEPRIV(R11)
                     00FA   281 RESTORE_PRIVS:
          7E  01  CE 00FA   282          MNEGL   #1,-(SP)        ;FORM MASK OF ALL PRIVS FOR DISABLE
          7E  01  CE 00FD   283          MNEGL   #1,-(SP)
          52  7E  7E 0100   284          MOVAQ   -(SP),R2        ;RESERVE 2ND MASK, R2 = ADR
                     0103   285          $SETPRV_S ENBFLG=#0,-   ;DISABLE ALL PROCESS PRIVILEGES
                     0103   286                    PRVADR=8(R2),-
                     0103   287                    PRVPRV=(R2)    ;SAVING OLD COPY
                     0113   288          $SETPRV_S ENBFLG=#1,-   ;ENABLE THE SAVED PRIVILEGES
                     0113   289                    PRVADR=PRC_Q_SAVEPRIV(R11)
00E8 CB  62  7D      0124   290          MOVQ    (R2),PRC_Q_SAVEPRIV(R11) ;SAVE PREVIOUS PRIVILEGES
     5E  10  C0      0129   291          ADDL    #16,SP          ;CLEAN OFF 2 PRIV MASKS
             05      012C   292          RSB
```

```
                        012D   294          .SBTTL  RUN DOWN IMAGE AND INDIRECT LEVELS
                        012D   295 ;+
                        012D   296 ; DCL$RUNDOWN - RUN DOWN IMAGE AND INDIRECT LEVELS
                        012D   297 ;
                        012D   298 ; THIS SUBROUTINE IS CALLED TO CHECK WHETHER INDIRECT LEVELS SHOULD BE RUN DOWN
                        012D   299 ; AND TO CLOSE RMS-32 FILES AND RUN DOWN THE PREVIOUS IMAGE.
                        012D   300 ;
                        012D   301 ; INPUTS:
                        012D   302 ;
                        012D   303 ;       NONE.
                        012D   304 ;
                        012D   305 ; OUTPUTS:
                        012D   306 ;
                        012D   307 ;       IF THE CURRENT LEVEL IS CONTROL Y/C, THEN ALL INDIRECT FILES ARE UNSTACKED.
                        012D   308 ;       IF THE PREVIOUS MODE WAS USER, THEN THE USER IMAGE EXIT HANDLERS ARE
                        012D   309 ;       EXECUTED. THE PREVIOUS IMAGE IS ALWAYS RUNDOWN.
                        012D   310 ;-
                        012D   311          .ENABL  LSB
                        012D   312
                        012D   313 DCL$RUNDWNI::                            ;RUN DOWN BUT PRESERVE INDIRECT LEVEL
          50   A0'AF   9E   012D   314          MOVAB   B^20$,R0               ;SET EXIT HANDLER RETURN ADDRESS
                 04   11   0131   315          BRB     5$                     ;
                        0133   316
                        0133   317 DCL$RUNDOWN::                            ;RUN DOWN IMAGE AND INDIRECT LEVELS
          50   96'AF   9E   0133   318          MOVAB   B^10$,R0               ;SET EXIT HANDLER RETURN ADDRESS
                        0137   319 5$:       SETBIT  PRC_V_DISABL,PRC_W_FLAGS(R11) ;DISABLE CONTROL Y/C AST'S
        60 68 AB    0B   E5   0138   320          BBCC    #PRC_V_YLEVEL,PRC_W_FLAGS(R11),20$ ;IF CLR, NOT AT CONTROL Y/C LEVEL
                 50   DD   013B   321          PUSHL   R0                     ;PUSH PROPER RETURN ADDRESS
              50   6B   D0   013D   322          MOVL    PRC_L_SAVAP(R11),R0    ;GET ADDRESS OF PREVIOUS PSL
           4D 60   18   E1   0140   323          BBC     #PSL$V_CURMOD,(R0),10$ ;IF CLR, PREVIOUS MODE SUPERVISOR
        03 A0    C8   8F   8A   0144   324          BICB    #<PSL$M_FPD!PSL$M_TP!PSL$M_CM>@-24,3(R0) ; RESET BITS IN PSL
     70   00000000'9F   9E   0149   325          MOVAB   @#EXE$EXIT_IMAGE,=(R0) ;RESET USER RETURN ADDRESS
      F8 A0    0980   8F   3C   0150   326          MOVZWL  #SS$_CLIFRCEXT,-8(R0)  ;SET EXIT CAUSE INTO SAVED R0
      F4 AA    5E     5A   C3   0156   327          SUBL3   R10,SP,WRK_L_SAVSP(R10) ;SAVE RELATIVE ADDRESS OF TOP OF STACK
           58   F8 AA   7D   015C   328          MOVQ    WRK_L_SAVAP(R10),R8    ;RETRIEVE PREVIOUS ARGUMENT AND FRAME POINTE
              6B   58   7D   0160   329          MOVQ    R8,PRC_L_SAVAP(R11)    ;SAVE IN PROCESS WORK AREA
           BA AA   5A   C2   0163   330          SUBL    R10,WRK_L_RSLNXT(R10)  ;CONVERT PARSE POINTER TO RELATIVE ADDRESS
           B6 AA   5A   C2   0167   331          SUBL    R10,WRK_L_RSLEND(R10)  ;CONVERT END POINTER TO RELATIVE ADDRESS
           57   08 A0   9E   016B   332          MOVAB   8(R0),R7               ;GET ADDRESS OF END OF ARGUMENT LIST + 4
              57   5D   C2   016F   333          SUBL    FP,R7                  ;CALCULATE LENGTH OF CALL FRAME AND ARGLIST
              5E   57   C2   0172   334          SUBL    R7,SP                  ;CALCULATE NEW TOP OF STACK ADDRESS
        6E    6D   57   28   0175   335          MOVC    R7,(FP),(SP)           ;MOVE CALL FRAME AND ARGUMENT LIST
        57    5D   5E   C3   0179   336          SUBL3   SP,FP,R7               ;CALCULATE LENGTH OF COMMAND BUFFER AND ARGL
        5D    59   57   C3   017D   337          SUBL3   R7,R9,FP               ;CALCULATE NEW TOP OF STACK ADDRESS
        6D    6E   57   28   0181   338          MOVC    R7,(SP),(FP)           ;COLLAPSE STACK REMOVING FIRST COMMAND CONTE
           F4 A9   59   C0   0185   339          ADDL    R9,WRK_L_SAVSP(R9)     ;CALCULATE NEW COMMAND STACK POINTER
           BA A9   59   C0   0189   340          ADDL    R9,WRK_L_RSLNXT(R9)    ;CONVERT PARSE POINTER TO REAL ADDRESS
           B6 A9   59   C0   018D   341          ADDL    R9,WRK_L_RSLEND(R9)    ;CONVERT END POINTER TO REAL ADDRESS
                 04   0191   342          RET                            ;RETURN TO EXE$EXIT_IMAGE
                        0195   343                                         ;  THEN TO 10$ OR 20$
           5C AB   D5   0196   344 10$:     TSTL    PRC_L_INDEPTH(R11)     ;INDIRECT LEVEL ZERO?
              05   13   0199   345          BEQL    20$                    ;IF EQL YES
            FE62'  30   019B   346          BSBW    DCL$UNSTACK            ;UNSTACK INDIRECT LEVEL
              F6   11   019E   347          BRB     10$                    ;
     0E 78 AB    00   E5   01A0   348 20$:     BBCC    #PRC_V_IRUNDWN,PRC_B_IMGFLAG(R11),30$ ;SKIP IF IMAGE ALREADY RUNDOWN
                        01A5   349          $RUNDWN_S #PSL$C_USER          ;RUN DOWN IMAGE (THE HARD WAY)
                 18   8A   01AE   350          BICB    #<PRC_M_EXEONLY ! PRC_M_PRIV>,- ;SINCE IMAGE IS NOW GONE
```

```
00AF CB        01B0   351                  PRC_B_FLAGS2(R11)       ;NO NEED TO PROTECT IT
          05   01B3   352 30$:    RSB                             ;
               01B4   353          .DSABL  LSB
```

```
                        01B4    355          .SBTTL  SHUT DOWN IMAGE
                        01B4    356 ;+
                        01B4    357 ; DCL$SHUTDOWN - SHUT DOWM IMAGE
                        01B4    358 ;
                        01B4    359 ; THIS ROUTINE IS CALLED TO CLOSE ALL FILES OPENED BY THE JUST EXECUTED IMAGE
                        01B4    360 ; AND TO CLOSE THE IMAGE ACTIVATION FILE.
                        01B4    361 ;
                        01B4    362 ; INPUTS:
                        01B4    363 ;
                        01B4    364 ;       R10 = BASE ADDRESS OF COMMAND WORK AREA.
                        01B4    365 ;       R11 = BASE ADDRESS OF PROCESS WORK AREA.
                        01B4    366 ;
                        01B4    367 ; OUTPUTS:
                        01B4    368 ;
                        01B4    369 ;       ALL FILES OPENED BY THE JUST EXECUTED IMAGE ARE CLOSED BY CALLING RMS-32,
                        01B4    370 ;       DATA RECORDS ARE SKIPPED IN THE INPUT STREAM, AND THE IMAGE FILE IS CLOSED.
                        01B4    371 ;       IF ANY DATA RECORDS ARE SKIPPED, THE INPUT BUFFER POINTER IS ADJUSTED
                        01B4    372 ;       AND EOL SET FOR THE LAST RECORD READ (FOR POSSIBLE FUTURE DCL$FLUSH
                        01B4    373 ;       OPERATION).
                        01B4    374 ;
                        01B4    375 ;               R4 DESTROYED
                        01B4    376 ;               R2 = NUMBER OF DATA RECORDS SKIPPED IN THE INPUT STREAM.
                        01B4    377 ;-
                        01B4    378
                        01B4    379 DCL$SHUTDOWN::                                      ;SHUT DOWN IMAGE
          48    10    01B4    380          BSBB    DCL$RMSRUNDWN                      ;RUNDOWN RMS-32 FILES
    53    14  AB    D0    01B6    381          MOVL    PRC_L_INDINPRAB(R11),R3            ;GET ADDRESS OF INDIRECT RAB
    7E    10  A3    7D    01BA    382          MOVQ    RAB$W_RFA(R3),-(SP)                ;SAVE RFA OF LAST COMMAND
 36 18  A3    00    E0    01BE    383          BBS     #DEV$V_REC,RAB$L_CTX(R3),30$       ;IF SET, RECORD ORIENTED DEVICE
          03    93    01C3    384          BITB    #PRC_M_CHAIN!PRC_M_CMD,-           ;CHAIN A/O COMMAND?
      00AF  CB          01C5    385                  PRC_B_FLAGS2(R11)                  ;
          2F    12    01C8    386          BNEQ    30$                                ;NO SKIP IF EITHER IS PENDING
                01CA    387          SETBIT  RAB$V_PPF_IND,RAB$W_ISI(R3)        ;CONVERT TO NONPRIVILEGED ISI
          52    D6    01CF    388 20$:     INCL    R2                                 ;INCREMENT NUMBER OF RECORDS SKIPPED
                01D1    389          $GET    RAB=(R3)                           ;GET NEXT RECORD FROM INDIRECT FILE
       06 50    E9    01DA    390          BLBC    R0,25$                             ;IF LBC FINISHED
    54 22  A3    3C    01DD    391          MOVZWL  RAB$W_RSZ(R3),R4                   ;SAVE LENGTH OF RECORD
          EC    11    01E1    392          BRB     20$
                01E3    393
                01E3    394 25$:     CLRBIT  RAB$V_PPF_IND,RAB$W_ISI(R3)        ;CONVERT BACK TO PRIVILEGED ISI
          52    D7    01E8    395          DECL    R2                                 ;ADJUST FOR LAST RECORD
          0D    13    01EA    396          BEQL    30$                                ;SKIP IF NO RECORDS READ IN
    54 28 B344  9E    01EC    397          MOVAB   @RAB$L_RBF(R3)[R4],R4              ;GET ADDR. OF END OF LAST RECORD
          64    94    01F1    398          CLRB    (R4)                               ;SET EOL IN BUFFER
F48E CA   FF    A4    9E    01F3    399          MOVAB   -1(R4),WRK_L_CHARPTR(R10)          ;ADJUST 'GET CHARACTER' POINTER
                01F9    400
    10 A3   8E    7D    01F9    401 30$:     MOVQ    (SP)+,RAB$W_RFA(R3)                ;RESTORE RFA OF LAST COMMAND
                05    01FD    402          RSB                                        ;
                01FE    403
```

```
                        01FE    405          .SBTTL  RMS RUNDOWN AN IMAGE
                        01FE    406 ;+
                        01FE    407 ; DCL$RMSRUNDWN - RMS RUNDOWN AN IMAGE
                        01FE    408 ;
                        01FE    409 ; THIS ROUTINE IS CALLED TO CLOSE ALL FILES OPENED BY THE JUST EXECUTED IMAGE
                        01FE    410 ; AND TO CLOSE THE IMAGE ACTIVATION FILE.
                        01FE    411 ;
                        01FE    412 ; INPUTS:
                        01FE    413 ;
                        01FE    414 ;       NONE
                        01FE    415 ;
                        01FE    416 ; OUTPUTS:
                        01FE    417 ;
                        01FE    418 ;       ALL FILES OPENED BY THE JUST EXECUTED IMAGE ARE CLOSED BY CALLING RMS-32,
                        01FE    419 ;       DATA RECORDS ARE SKIPPED IN THE INPUT STREAM, AND THE IMAGE FILE IS CLOSED.
                        01FE    420 ;
                        01FE    421 ;-
                        01FE    422
                        01FE    423 DCL$RMSRUNDWN::                                     ;RMS RUNDOWN THE IMAGE
          FDFF'  30     01FE    424          BSBW    DCL$ALLOCBUF                       ;ALLOCATE BUFFER AND DESCRIPTOR
    62  84 8F   9A     0201    425 10$:     MOVZBL  #WRK_C_MSGBUFSIZ,(R2)              ;RESET SIZE OF MESSAGE BUFFER
          00    DD     0205    426          PUSHL   #0                                 ;RUN DOWN ONLY IMAGE FILES
          62    9F     0207    427          PUSHAB  (R2)                               ;SET ADDRESS OF MESSAGE BUFFER DESCR
00000000'9F     02    FB 0209    428          CALLS   #2,@#SYS$RMSRUNDWN                 ;RUNDOWN RMS-32 FILES
       EE 50    E9     0210    429          BLBC    R0,10$                             ;IF RUNDOWN FAILURE CONTINUE WITH NE
    5E  008C CE     9E     0213    430          MOVAB   WRK_C_MSGBUFSIZ+8(SP),SP           ;DEALLOCATE MESSAGE BUFFER AND DESCR
          52    D4     0218    431          CLRL    R2                                 ;CLEAR COUNT OF RECORDS SKIPPED
          05    021A    432          RSB                                                ;RETURN
                        021B    433
                        021B    434          .END
```

Symbol table

```
$$.TMP1                 =  00000001
$$.TMP2                 =  00000063
CLI$_IVVALU             =  00038088
CLI$_NORMAL             =  00030001
CTL$GL_CLINTOWN            ********    X   02
CTL$GL_DCLPRSOWN           ********    X   02
DCL$ABORT                  ********    X   02
DCL$ALLOCBUF               ********    X   02
DCL$CONTINUE               00000010 RG     02
DCL$DEBUG                  0000001D RG     02
DCL$GETDVAL                ********      X 02
DCL$RMSRUNDWN              000001FE RG     02
DCL$RUNDOWN                00000133 RG     02
DCL$RUNDWNI                0000012D RG     02
DCL$SAVE_PRIVS             000000E9 RG     02
DCL$SHUTDOWN               000001B4 RG     02
DCL$STOP                   0000003F RG     02
DCL$UNSTACK                ********      X 02
DEV$V_REC               =  00000000
EXE$EXIT_IMAGE             ********    X   02
EXE$REFLECT                ********    X   02
HEXTAB                     00000000 R      02
PPD$B_NPROCS               0000001C
PPD$C_LENGTH               00000168
PPD$K_LENGTH               00000168
PPD$L_INPDEV               00000044
PPD$L_LGI                  00000014
PPD$L_LSTSTATUS            00000018
PPD$L_OUTDEV               00000064
PPD$L_PRC                  00000008
PPD$Q_CLIREG               00000004
PPD$Q_CLISYMTBL            0000000C
PPD$T_FILENAME             00000068
PPD$T_INPDVI               00000028
PPD$T_OUTDVI               00000048
PPD$W_FLAGS                00000002
PPD$W_INPCHAN              0000001E
PPD$W_INPDID               0000003E
PPD$W_INPFID               00000038
PPD$W_INPIFI               00000020
PPD$W_INPISI               00000022
PPD$W_OUTDID               0000005E
PPD$W_OUTFID               00000058
PPD$W_OUTIFI               00000024
PPD$W_OUTISI               00000026
PPD$W_SIZE                 00000000
PRC_B_CONTINUE             000000F3
PRC_B_DEFRADIX             000000AE
PRC_B_EXMDEPMOD            000000AD
PRC_B_EXMDEPWID            000000AC
PRC_B_EXONLYL              0000012D
PRC_B_FLAGS2               000000AF
PRC_B_IMGFLAG              00000078
PRC_B_OUTFLAGS             0000012C
PRC_B_PROMPTLEN            000000F0
PRC_C_LENGTH               00000534
PRC_G_COMMANDS             00000133
PRC_G_PROMPT               000000F4
PRC_K_LENGTH               00000534
PRC_L_CURRKEY              00000048
PRC_L_EXMDEPADR            000000A8
PRC_L_EXTARG               00000094
PRC_L_EXTBLK               0000008C
PRC_L_EXTCOD               0000009C
PRC_L_EXTHND               00000090
PRC_L_EXTPRM               00000098
PRC_L_IDFLNK               000000BC
PRC_L_IMGACTSTS            00000080
PRC_L_INDCLOCK             0000007C
PRC_L_INDEPTH              0000005C
PRC_L_INDFAB               0000001C
PRC_L_INDINPRAB            00000014
PRC_L_INDOUTRAB            00000018
PRC_L_INPRAB               00000008
PRC_L_LASTKEY              0000004C
PRC_L_LSTSTATUS            000000B0
PRC_L_ONCTLY               000000B8
PRC_L_ONERROR              0000006C
PRC_L_OUTOFBAND            000000B4
PRC_L_OUTRAB               0000000C
PRC_L_OUTRABCTX            00000118
PRC_L_PPFLIST              00000070
PRC_L_RECALLPTR            0000012F
PRC_L_RESTART              00000058
PRC_L_SAVAP                00000000
PRC_L_SAVFP                00000004
PRC_L_SEVERITY             00000050
PRC_L_SPWN                 000000C0
PRC_L_STACKLM              000000A4
PRC_L_STACKPT              000000A0
PRC_L_STATUS               00000054
PRC_L_STS                  00000084
PRC_L_STV                  00000088
PRC_L_SYMBOL               00000060
PRC_L_TMBX                 00000074
PRC_L_TRMLIST              00000010
PRC_M_CHAIN             =  00000002
PRC_M_CMD               =  00000001
PRC_M_EXEONLY           =  00000008
PRC_M_PRIV              =  00000010
PRC_Q_ALLOCREG             00000020
PRC_Q_COMMAND              000000E0
PRC_Q_FLUSHTIME            000000D0
PRC_Q_GLOBAL               00000028
PRC_Q_IMAGENAME            000000D8
PRC_Q_KEYPAD               00000040
PRC_Q_LABEL                00000030
PRC_Q_LOCAL                00000038
PRC_Q_SAVEPRIV             000000E8
PRC_T_OUTDVI               0000011C
PRC_V_DISABL            =  00000002
PRC_V_IRUNDWN           =  00000000
PRC_V_MODE              =  00000006
PRC_V_PRIV              =  00000004
```

```
PRC_V_YLEVEL            =  0000000B
PRC_W_ASTIOSB              000000C6
PRC_W_ASTRETN              000000C8
PRC_W_ASTSTATUS            000000C4
PRC_W_ATTMBX               0000007A
PRC_W_FLAGS                00000068
PRC_W_INPCHAN              00000064
PRC_W_ONLEVEL              0000006A
PRC_W_OUTIFI               00000114
PRC_W_OUTISI               00000116
PRC_W_OUTMBXCHN            000000CA
PRC_W_OUTMBXREF            000000CE
PRC_W_OUTMBXSIZ            000000CC
PRC_W_PMPTCTRL             000000F1
PRC_W_WAITIOSB             00000066
PSL$C_SUPER             =  00000002
PSL$C_USER              =  00000003
PSL$M_CM                =  80000000
PSL$M_FPD               =  08000000
PSL$M_TP                =  40000000
PSL$V_CURMOD            =  00000018
PSL$V_PRVMOD            =  00000016
PTR_B_LEVEL                00000004
PTR_B_NUMBER               00000005
PTR_B_PARMCNT              00000006
PTR_B_VALUE                00000000
PTR_C_LENGTH               0000000C
PTR_K_COMDQUAL          =  00000000
PTR_K_ENDLINE           =  00000004
PTR_K_LENGTH               0000000C
PTR_L_DESCR                00000000
PTR_L_ENTITY               00000008
RAB$L_CTX               =  00000018
RAB$L_RBF               =  00000028
RAB$V_PPF_IND           =  0000000E
RAB$W_ISI               =  00000002
RAB$W_RFA               =  00000010
RAB$W_RSZ               =  00000022
RESTORE_PRIVS              000000FA R      02
SS$_CLIFRCEXT           =  00000980
SS$_DEBUG               =  0000046C
SYS$DELPRC                 ********  GX    02
SYS$GET                    ********  GX    02
SYS$RMSRUNDWN              ********   X    02
SYS$RUNDWN                 ********  GX    02
SYS$SETPRV                 ********  GX    02
TESTMODE                   000000CC R      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_C_MSGBUFSIZ         =  00000084
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_W_FLAGS                FFFFFFF0
WRK_W_FLAGS2               FFFFFFF2
WRK_W_IMGCHAN              FFFFFFEE
WRK_W_PMPTLEN              FFFFF99E
_$$_                    =  000000EF
```

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	0000021B (539.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	11	00:00:00.05	00:00:00.97
Command processing	88	00:00:00.70	00:00:06.64
Pass 1	318	00:00:12.28	00:00:43.86
Symbol table sort	0	00:00:01.66	00:00:06.20
Pass 2	85	00:00:02.20	00:00:08.02
Symbol table output	23	00:00:00.20	00:00:00.72
Psect synopsis output	2	00:00:00.03	00:00:00.10
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	527	00:00:17.12	00:01:06.52

The working set limit was 1350 pages.
62782 bytes (123 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1143 non-local and 19 local symbols.
434 source lines were read in Pass 1, producing 14 object records in Pass 2.
43 pages of virtual memory were used to define 28 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]SYSBLDMLB.MLB;1	0
_255DUA28:[DCL.OBJ]DCL.MLB;1	9
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	13
TOTALS (all libraries)	22

1370 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

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

GETKEYNAM
LIS

GOTO
LIS

EXPRESS
LIS

HANDLE
LIS

IMAGECTRL
LIS

INITIAL
LIS

INDIRECT
LIS

FILECMDS
LIS

IF
LIS

IMAGEXECT
LIS