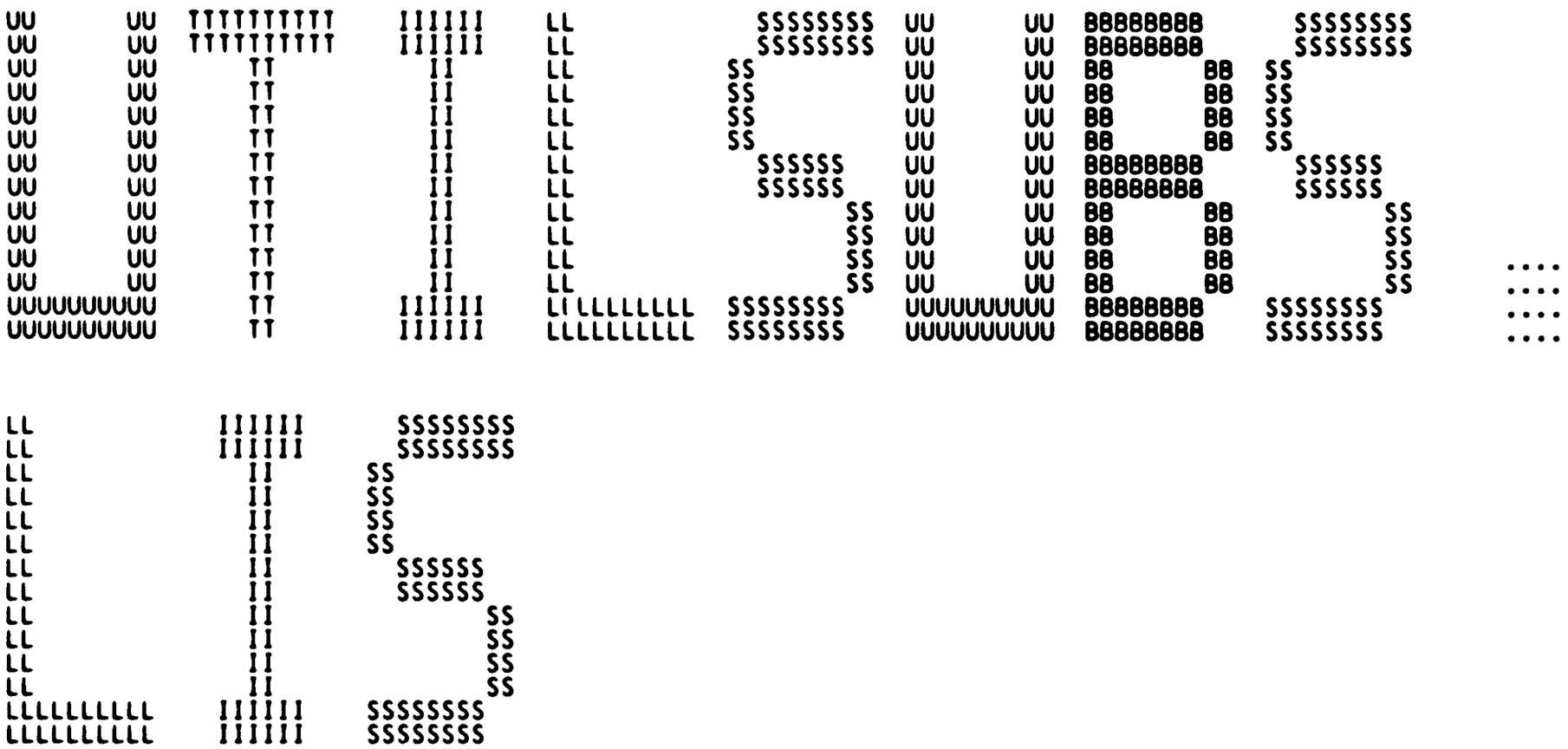
```
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL                 IIIIIIIII     UUU         UUU  TTTTTTTTTTTTTTT  LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
CCC              LLL                    III        UUU         UUU        TTT        LLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL     IIIIIIIII     UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
```

FILEID**UTILSUBS



```
;     1          0001  0 MODULE utilsubs ( IDENT = 'V04-000',
;     2          0002  0                  ADDRESSING_MODE (EXTERNAL = GENERAL,
;     3          0003  0                                   NONEXTERNAL = LONG_RELATIVE)) =
;     4          0004  1 BEGIN
;     5          0005  1
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:  SET and SHOW Command utility routines
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains support routines to manipulate
;    36          0036  1 !       UICs, as well as file and device protections.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:
;    39          0039  1 !
;    40          0040  1 !       VAX/VMS operating system, user mode
;    41          0041  1 !
;    42          0042  1 ! AUTHOR:  Gerry Smith                              23-Feb-1983
;    43          0043  1 !
;    44          0044  1 ! Modified by:
;    45          0045  1 !
;    46          0046  1 !       V03-004 LMP0191         L. Mark Pilant,         13-Feb-1984  9:29
;    47          0047  1 !               Return the status from the UIC parse.
;    48          0048  1 !
;    49          0049  1 !       V03-003 GAS0179         Gerry Smith              7-Sep-1983
;    50          0050  1 !               Make non-external references longword references instead
;    51          0051  1 !               of word references.
;    52          0052  1 !
;    53          0053  1 !       V03-002 LMP0140         L. Mark Pilant,         23-Aug-1983  10:10
;    54          0054  1 !               Add support for alphanumeric UICs.
;    55          0055  1 !
;    56          0056  1 !       V03-001 GAS0119         Gerry Smith             14-Apr-1983
;    57          0057  1 !               Fix sense of KEY in EXPAND_PROT.
```

```
;   58          0058  1 !
;   59          0059  1 !--
```

```
;    61          0060  1 !
;    62          0061  1 ! Include files
;    63          0062  1 !
;    64          0063  1 LIBRARY 'SYS$LIBRARY:STARLET';                  ! VAX/VMS common definitions
;    65          0064  1 LIBRARY 'SYS$LIBRARY:TPAMAC';                   ! TPARSE macros
;    66          0065  1
;    67          0066  1
```

```
;    69          0067  1 !
;    70          0068  1 ! Table of contents
;    71          0069  1 !
;    72          0070  1
;    73          0071  1 FORWARD ROUTINE
;    74          0072  1     parse_uic,                      ! Convert ASCII UIC to longword
;    75          0073  1     get_prot : NOVALUE,             ! Convert ASCII protection to binary
;    76          0074  1     parse_class,                    ! Parse one class of user protection
;    77          0075  1     expand_prot : NOVALUE;          ! Convert binary prot to ASCII
;    78          0076  1
;    79          0077  1 !
;    80          0078  1 ! External routines
;    81          0079  1 !
;    82          0080  1 EXTERNAL ROUTINE
;    83          0081  1     cli$present,                    ! Detect presence of value from CLI
;    84          0082  1     cli$get_value,                  ! Get value from CLI
;    85          0083  1     LIB$TPARSE;                     ! General purpose parser
;    86          0084  1
;    87          0085  1 !
;    88          0086  1 ! External symbols
;    89          0087  1 !
;    90          0088  1 EXTERNAL LITERAL
;    91          0089  1     cli$_ivprot;                    ! Invalid protection specification
;    92          0090  1
;    93          0091  1 OWN
;    94          0092  1         CONVERTED_UIC;              ! Converted UIC value
;    95          0093  1
;    96          0094  1 !
;    97          0095  1 ! Parse the UIC string and store the binary value.
;    98          0096  1 !
;    99          0097  1 $INIT_STATE     (UIC_STB, UIC_KTB);
;   100          0098  1
;   101        P 0099  1 $STATE  (,
;   102        P 0100  1          (TPA$_IDENT,,,,CONVERTED_UIC)
;   103          0101  1          );
;   104        P 0102  1 $STATE  (,
;   105        P 0103  1          (TPA$_EOS,TPA$_EXIT)
;   106          0104  1          );
```

```
;  108          0105  1 GLOBAL ROUTINE parse_uic (desc, uic) =
;  109          0106  2 BEGIN
;  110          0107  2
;  111          0108  2 !---
;  112          0109  2 !
;  113          0110  2 ! This routine takes an ASCII string of the form [m,n] and attempts to parse
;  114          0111  2 ! the pieces into a longword UIC.  If any errors are detected, an error is
;  115          0112  2 ! returned.
;  116          0113  2 !
;  117          0114  2 ! Inputs
;  118          0115  2 !       DESC - address of ASCII descriptor of UIC string
;  119          0116  2 !
;  120          0117  2 ! Outputs
;  121          0118  2 !       UIC  - the longword representation is returned here.
;  122          0119  2 !
;  123          0120  2 !---
;  124          0121  2
;  125          0122  2 MAP
;  126          0123  2     desc : REF $BBLOCK,
;  127          0124  2     uic : REF VECTOR;
;  128          0125  2
;  129          0126  2 BIND string = .desc[dsc$a_pointer] : VECTOR[,BYTE];
;  130          0127  2
;  131          0128  2 LOCAL
;  132          0129  2     TPARSE_BLOCK        : $BBLOCK [TPA$K_LENGTH0]       ! TPARSE
;  133          0130  2                           INITIAL (TPA$K_COUNTO,        !  PARAMETER
;  134          0131  2                                    TPA$M_BLANKS OR      !  BLOCK
;  135          0132  2                                    TPA$M_ABBREV),
;  136          0133  2     STATUS;                                             ! Routine return status
;  137          0134  2
;  138          0135  2 TPARSE_BLOCK[TPA$L_STRINGCNT] = .DESC[DSC$W_LENGTH];
;  139          0136  2 TPARSE_BLOCK[TPA$L_STRINGPTR] = .DESC[DSC$A_POINTER];
;  140          0137  2
;  141          0138  3 IF NOT (STATUS = LIB$TPARSE (TPARSE_BLOCK, UIC_STB, UIC_KTB))
;  142          0139  2 THEN RETURN .STATUS;
;  143          0140  2
;  144          0141  2 UIC[0] = .CONVERTED_UIC;                        ! MAKE THE UIC LONGWORD
;  145          0142  2
;  146          0143  2 RETURN 1;
;  147          0144  1 END;


                                                .TITLE  UTILSUBS
                                                .IDENT  \V04-000\

                                                .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                                    00000 UIC_STB::
                                                .BLKB   0
                                45EC  00000 ;TPA$TYPE
                                            U.2:    .WORD   17900                                   ;
                            00000000* 00002 ;TPA$ADDR
                                            U.3:    .LONG   <<CONVERTED_UIC-U.3>-4>                 ;
                                15F7  00006 ;TPA$TYPE
                                            U.4:    .WORD   5623                                    ;
                                FFFF  00008 ;TPA$TARGET
                                            U.5:    .WORD   -1                                      ;
```

```
                                                        .PSECT  _LIB$KEY0$,NOWRT,  SHR,  PIC,1

                                                00000 UIC_KTB::
                                                        .BLKB   0
                                                00000 ;TPA$KEY0
                                                      U.1:      .BLKB   0

                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                00000003  00000008  00000 P.AAA:  .LONG   8, 3                                    ;

                                                        .PSECT  $OWN$,NOEXE,2

                                                00000 CONVERTED_UIC:
                                                        .BLKB   4

                                                        .EXTRN  CLI$PRESENT, CLI$GET_VALUE
                                                        .EXTRN  LIB$TPARSE, CLI$_IVPROT

                                                        .PSECT  $CODE$,NOWRT,2

                                                    007C 00000          .ENTRY  PARSE_UIC, Save R2,R3,R4,R5,R6          ; 0105
                                        5E              24   C2 00002          SUBL2   #36, SP                         ;
                                        56        04    AC   D0 00005          MOVL    DESC, R6                        ; 0126
                24              00 00000000'    EF          08   2C 00009          MOVC5   #8, P.AAA, #0, #36, TPARSE_BLOCK ; 0132
                                                            6E   00012          ;
                                  08    AE              66   3C 00013          MOVZWL  (R6), TPARSE_BLOCK+8            ; 0135
                                  0C    AE        04    A6   D0 00017          MOVL    4(R6), TPARSE_BLOCK+12          ; 0136
                                              00000000' EF   9F 0001C          PUSHAB  UIC_KTB                         ; 0138
                                              00000000' EF   9F 00022          PUSHAB  UIC_STB                         ;
                                                    08  AE   9F 00028          PUSHAB  TPARSE_BLOCK                    ;
                                00000000G       00          03   FB 0002B          CALLS   #3, LIB$TPARSE                  ;
                                                08          50   E9 00032          BLBC    STATUS, 1$                      ;
                                      08        BC 00000000'  EF   D0 00035          MOVL    CONVERTED_UIC, @UIC             ; 0141
                                                50          01   D0 0003D          MOVL    #1, R0                          ; 0143
                                                                 04 00040 1$:     RET                                     ; 0144

; Routine Size:  65 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  149          0145  1 GLOBAL ROUTINE get_prot (class, prot) : NOVALUE =
;  150          0146  2 BEGIN
;  151          0147  2
;  152          0148  2 !---
;  153          0149  2 !
;  154          0150  2 ! This routine interrogates the CLI to get the requested protection, and then
;  155          0151  2 ! converts that ASCII representation to a word of binary data which is the
;  156          0152  2 ! representation of protection used by the system.
;  157          0153  2 !
;  158          0154  2 ! Inputs:
;  159          0155  2 !       None.
;  160          0156  2 !
;  161          0157  2 ! Outputs:
;  162          0158  2 !
;  163          0159  2 !       CLASS - a word bitmask indicating what classes of protection were
;  164          0160  2 !               referenced.
;  165          0161  2 !       PROT  - a word bitmask representing the requested protection
;  166          0162  2 !
;  167          0163  2 !---
;  168          0164  2
;  169          0165  2 LOCAL
;  170          0166  2     status,
;  171          0167  2     desc : $BBLOCK[dsc$c_s_bln];        ! Descriptor
;  172          0168  2
;  173          0169  2 MAP
;  174          0170  2     class : REF VECTOR[,WORD],
;  175          0171  2     prot : REF VECTOR[,WORD];
;  176          0172  2
;  177          0173  2 !
;  178          0174  2 ! Define the tables of ASCII descriptors, to be used by the CLI.
;  179          0175  2 !
;  180          0176  2 OWN class_table : VECTOR[4] INITIAL(%ASCID 'OPTION.PROTECTION.SYSTEM',
;  181          0177  2                                    %ASCID 'OPTION.PROTECTION.OWNER',
;  182          0178  2                                    %ASCID 'OPTION.PROTECTION.GROUP',
;  183          0179  2                                    %ASCID 'OPTION.PROTECTION.WORLD');
;  184          0180  2
;  185          0181  2
;  186          0182  2 !
;  187          0183  2 ! Initialize the descriptor.  Clear the mask and protection word.
;  188          0184  2 !
;  189          0185  2 $init_dyndesc(desc);
;  190          0186  2 class[0] = prot[0] = 0;
;  191          0187  2
;  192          0188  2 !
;  193          0189  2 ! For each class (SYSTEM, OWNER, GROUP, and WORLD), see if a protection
;  194          0190  2 ! was given.  If so, parse that into a binary representation.
;  195          0191  2 !
;  196          0192  2 INCR index FROM 0 TO 3 DO
;  197          0193  3     BEGIN
;  198          0194  3     IF cli$present(.class_table[.index])
;  199          0195  3     THEN
;  200          0196  4         BEGIN
;  201          0197  4         class[0] = .class[0] OR ((%X'F')^(.index*4));
;  202          0198  4         IF cli$get_value(.class_table[.index], desc)
;  203          0199  4         THEN prot[0] = .prot[0] OR (parse_class(desc)^(.index*4));
;  204          0200  3         END;
;  205          0201  2     END;
```

```
;  206          0202  2
;  207          0203  2 !
;  208          0204  2 ! Complement the protection value since at this point, a bit set true
;  209          0205  2 ! indicates that we want to ALLOW access, while the system convention
;  210          0206  2 ! is that a bit set true indicates that we want to DENY access.
;  211          0207  2 !
;  212          0208  2 IF .class[0] NEQ 0                          ! If any protections specified
;  213          0209  2 THEN prot[0] = NOT .prot[0];                ! then get the complement
;  214          0210  2
;  215          0211  2 RETURN 1;
;  216          0212  1 END;


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00008 P.AAC:  .ASCII  \OPTION.PROTECTION.SYSTEM\                ;
                  4D 45 54 53 59 53 2E 4E 4F  00017
                                 010E0018     00020 P.AAB:  .LONG   17694744                                 ;
                                 00000000'    00024         .ADDRESS P.AAC                                   ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00028 P.AAE:  .ASCII  \OPTION.PROTECTION.OWNER\<0>            ;
                  00 52 45 4E 57 4F 2E 4E 4F  00037
                                 010E0017     00040 P.AAD:  .LONG   17694743                                 ;
                                 00000000'    00044         .ADDRESS P.AAE                                   ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00048 P.AAG:  .ASCII  \OPTION.PROTECTION.GROUP\<0>            ;
                  00 50 55 4F 52 47 2E 4E 4F  00057
                                 010E0017     00060 P.AAF:  .LONG   17694743                                 ;
                                 00000000'    00064         .ADDRESS P.AAG                                   ;
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F  00068 P.AAI:  .ASCII  \OPTION.PROTECTION.WORLD\<0>            ;
                  00 44 4C 52 4F 57 2E 4E 4F  00077
                                 010E0017     00080 P.AAH:  .LONG   17694743                                 ;
                                 00000000'    00084         .ADDRESS P.AAI                                   ;

                                                                    .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00004 CLASS_TABLE:
                                                                    .ADDRESS P.AAB, P.AAD, P.AAF, P.AAH         ;


                                                                    .PSECT  $CODE$,NOWRT,2

                                    000C 00000          .ENTRY  GET_PROT, Save R2,R3                        ; 0145
                 53 00000000'   EF  9E 00002          MOVAB   CLASS_TABLE, R3                             ;
                 5E             04  C2 00009          SUBL2   #4, SP                                      ;
                    020E0000    8F  DD 0000C          PUSHL   #34471936                                   ; 0185
                          04    AE  D4 00012          CLRL    DESC+4                                      ;
                          08    BC  B4 00015          CLRW    @PROT                                       ; 0186
                          04    BC  B4 00018          CLRW    @CLASS                                      ;
                                52  D4 0001B          CLRL    INDEX                                       ; 0192
                             6342   DD 0001D 1$:      PUSHL   CLASS_TABLE[INDEX]                          ; 0194
    00000000G    00              01  FB 00020          CALLS   #1, C[I$PRESENT                             ;
                 30              50  E9 00027          BLBC    R0, 2$                                      ;
51               52              02  78 0002A          ASHL    #2, INDEX, R1                               ; 0197
50               0F              51  78 0002E          ASHL    R1, #15, R0                                 ;
          04     BC              50  A8 00032          BISW2   R0, @CLASS                                  ;
                                 5E  DD 00036          PUSHL   SP                                          ; 0198
```

```
                                           6342  DD 00038          PUSHL   CLASS_TABLE[INDEX]              ;
                       00000000G  00         02  FB 0003B          CALLS   #2, C[I$GET_VALUE               ;
                                  15         50  E9 00042          BLBC    R0, 2$                          ;
                                             5E  DD 00045          PUSHL   SP                              ; 0199
                       00000000V  EF         01  FB 00047          CALLS   #1, PARSE_CLASS                 ;
                 51               52         02  78 0004E          ASHL    #2, INDEX, R1                   ;
                 50               50         51  78 00052          ASHL    R1, R0, R0                      ;
                            08    BC         50  A8 00056          BISW2   R0, @PROT                       ;
                 BF               52         03  F3 0005A 2$:      AOBLEQ  #3, INDEX, 1$                   ; 0192
                                        04   BC  B5 0005E          TSTW    @CLASS                          ; 0208
                                             05  13 00061          BEQL    3$                              ;
                            08    BC    08   BC  B2 00063          MCOMW   @PROT, @PROT                    ; 0209
                                                 04 00068 3$:      RET                                     ; 0212

; Routine Size:  105 bytes,    Routine Base:  $CODE$ + 0041
```

```
;  218          0213  1 ROUTINE parse_class (desc) =
;  219          0214  2 BEGIN
;  220          0215  2
;  221          0216  2 !---
;  222          0217  2 !
;  223          0218  2 ! This routine parses one class of user (e.g. SYSTEM, OWNER, GROUP, WORLD)
;  224          0219  2 ! to see what protection is allowed.  The value returned in the low 4 bits
;  225          0220  2 ! is the protection code, with the bits set to reflect that access is
;  226          0221  2 ! requested.  Note that this is exactly the opposite of what the system wants.
;  227          0222  2 !
;  228          0223  2 ! Inputs:
;  229          0224  2 !
;  230          0225  2 !       DESC - a descriptor pointing to the ASCII representation of the
;  231          0226  2 !              protection desired
;  232          0227  2 !
;  233          0228  2 !---
;  234          0229  2
;  235          0230  2 MAP desc : REF $BBLOCK;
;  236          0231  2
;  237          0232  2 LOCAL
;  238          0233  2     result,
;  239          0234  2     string : REF VECTOR[,BYTE];         ! String pointer
;  240          0235  2
;  241          0236  2 !
;  242          0237  2 ! Initially set the value to all zeros, no access
;  243          0238  2 !
;  244          0239  2 result = 0;
;  245          0240  2
;  246          0241  2 !
;  247          0242  2 ! Scan for the occurrence of each keyletter, and, if it is there, set the
;  248          0243  2 ! appropriate bit.
;  249          0244  2 !
;  250          0245  2 string = .desc[dsc$a_pointer];
;  251          0246  2 INCR index FROM 0 to (.desc[dsc$w_length] -1) DO
;  252          0247  3     BEGIN
;  253          0248  3     IF .string[.index] EQL'R'
;  254          0249  3     THEN result = .result OR %X'1'
;  255          0250  3     ELSE IF .string[.index] EQL  'W'
;  256          0251  3     THEN result = .result OR %X'2'
;  257          0252  3     ELSE IF .string[.index] EQL 'E'
;  258          0253  3          OR .string[.index] EQL 'P'
;  259          0254  3     THEN result = .result OR %X'4'
;  260          0255  3     ELSE IF .string[.index] EQL 'D'
;  261          0256  3          OR .string[.index] EQL 'L'
;  262          0257  3     THEN result = .result OR %X'8'
;  263          0258  3     ELSE SIGNAL_STOP(cli$_ivprot);
;  264          0259  2     END;
;  265          0260  2
;  266          0261  2 RETURN .result;
;  267          0262  1 END;


                                003C 00000 PARSE_CLASS:
                                           .WORD   Save R2,R3,R4,R5                    ; 0213
```

```
                                    54  D4 00002          CLRL    RESULT                                  ; 0239
                                50  04  AC  D0 00004          MOVL    DESC, R0                                ; 0245
                                52  04  A0  D0 00008          MOVL    4(R0), STRING
                                55      60  3C 0000C          MOVZWL  (R0), R5                                ; 0246
                                53      01  CE 0000F          MNEGL   #1, INDEX                               ; 0248
                                        49  11 00012          BRB     8$
                                50    6342  9A 00014 1$:      MOVZBL  (INDEX)[STRING], R0
                            52  8F      50  91 00018          CMPB    R0, #82
                                        05  12 0001C          BNEQ    2$
                                54      01  88 0001E          BISB2   #1, RESULT                              ; 0249
                                        3A  11 00021          BRB     8$
                            57  8F      50  91 00023 2$:      CMPB    R0, #87                                 ; 0250
                                        05  12 00027          BNEQ    3$
                                54      02  88 00029          BISB2   #2, RESULT                              ; 0251
                                        2F  11 0002C          BRB     8$
                            45  8F      50  91 0002E 3$:      CMPB    R0, #69                                 ; 0252
                                        06  13 00032          BEQL    4$
                            50  8F      50  91 00034          CMPB    R0, #80                                 ; 0253
                                        05  12 00038          BNEQ    5$
                                54      04  88 0003A 4$:      BISB2   #4, RESULT                              ; 0254
                                        1E  11 0003D          BRB     8$
                            44  8F      50  91 0003F 5$:      CMPB    R0, #68                                 ; 0255
                                        06  13 00043          BEQL    6$
                            4C  8F      50  91 00045          CMPB    R0, #76                                 ; 0256
                                        05  12 00049          BNEQ    7$
                                54      08  88 0004B 6$:      BISB2   #8, RESULT                              ; 0257
                                        0D  11 0004E          BRB     8$
                        00000000G   8F      DD 00050 7$:      PUSHL   #CLI$_IVPROT                            ; 0258
                    00000000G   00      01  FB 00056          CALLS   #1, LIB$STOP
                B3              53      55  F2 0005D 8$:      AOBLSS  R5, INDEX, 1$                           ; 0246
                                50      54  D0 00061          MOVL    RESULT, R0                              ; 0261
                                            04 00064          RET                                             ; 0262

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 00AA
```

```
;  269          0263  1 GLOBAL ROUTINE expand_prot (table, prot, key) : NOVALUE =
;  270          0264  2 BEGIN
;  271          0265  2
;  272          0266  2 !----
;  273          0267  2 !
;  274          0268  2 ! Functional description
;  275          0269  2 !
;  276          0270  2 !       This routine fills a given VECTOR with the addresses of
;  277          0271  2 !       strings corresponding to a given protection word.
;  278          0272  2 !
;  279          0273  2 ! Input parameters
;  280          0274  2 !
;  281          0275  2 !       TABLE - address of the table to be filled in
;  282          0276  2 !       PROT  - protection word
;  283          0277  2 !       KEY   - flag to indicate how to translate the protection
;  284          0278  2 !               0 => RWED (file protection)
;  285          0279  2 !               1 => RWPL (device protection)
;  286          0280  2 !
;  287          0281  2 ! Output parameters
;  288          0282  2 !
;  289          0283  2 !       TABLE has been filled in with the addresses of descriptors
;  290          0284  2 !       of strings describing each type of user (SYS,OWN,GRP,WORLD).
;  291          0285  2 !
;  292          0286  2 !----
;  293          0287  2
;  294          0288  2 BIND
;  295          0289  2     prot_table = .table: VECTOR[4];     ! Table of addresses
;  296          0290  2
;  297          0291  2 OWN
;  298          0292  2     rwed_values: VECTOR[16] INITIAL(%ASCID 'RWED',      ! File
;  299          0293  2                                     %ASCID 'WED',       ! protection
;  300          0294  2                                     %ASCID 'RED',       ! descriptions
;  301          0295  2                                     %ASCID 'ED',
;  302          0296  2                                     %ASCID 'RWD',
;  303          0297  2                                     %ASCID 'WD',
;  304          0298  2                                     %ASCID 'RD',
;  305          0299  2                                     %ASCID 'D',
;  306          0300  2                                     %ASCID 'RWE',
;  307          0301  2                                     %ASCID 'WE',
;  308          0302  2                                     %ASCID 'RE',
;  309          0303  2                                     %ASCID 'E',
;  310          0304  2                                     %ASCID 'RW',
;  311          0305  2                                     %ASCID 'W',
;  312          0306  2                                     %ASCID 'R',
;  313          0307  2                                     %ASCID ''),
;  314          0308  2
;  315          0309  2     rwpl_values: VECTOR[16] INITIAL(%ASCID 'RWLP',      ! Device
;  316          0310  2                                     %ASCID 'WLP',       ! protection
;  317          0311  2                                     %ASCID 'RLP',       ! descriptions
;  318          0312  2                                     %ASCID 'LP',
;  319          0313  2                                     %ASCID 'RWL',
;  320          0314  2                                     %ASCID 'WL',
;  321          0315  2                                     %ASCID 'RL',
;  322          0316  2                                     %ASCID 'L',
;  323          0317  2                                     %ASCID 'RWP',
;  324          0318  2                                     %ASCID 'WP',
;  325          0319  2                                     %ASCID 'RP',
```

```
;  326     0320  2                                    %ASCID 'P',
;  327     0321  2                                    %ASCID 'RW',
;  328     0322  2                                    %ASCID 'W',
;  329     0323  2                                    %ASCID 'R',
;  330     0324  2                                    %ASCID '');
;  331     0325  2
;  332     0326  2 !
;  333     0327  2 ! Based on the value of KEY, use the appropriate device protection description
;  334     0328  2 ! table.  For each class (SYSTEM, OWNER, GROUP, WORLD), use the 4-bit
;  335     0329  2 ! protection pattern as an index into the protection table.
;  336     0330  2 !
;  337     0331  2 IF .key EQL 0
;  338     0332  2 THEN INCR index FROM 0 TO 3 DO
;  339     0333  3     (prot_table[.index] = .rwed_values [.prot<.index*4,4>])
;  340     0334  2 ELSE INCR index FROM 0 TO 3 DO
;  341     0335  2     (prot_table[.index] = .rwpl_values [.prot<.index*4,4>]);
;  342     0336  2
;  343     0337  2 RETURN;
;  344     0338  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                44  45  57  52  00088 P.AAK:  .ASCII  \RWED\                    ;
                                  010E0004  0008C P.AAJ:  .LONG   17694724                     ;
                                  00000000' 00090         .ADDRESS P.AAK                       ;
                                00  44  45  57  00094 P.AAM:  .ASCII  \WED\<0>                  ;
                                  010E0003  00098 P.AAL:  .LONG   17694723                     ;
                                  00000000' 0009C         .ADDRESS P.AAM                       ;
                                00  44  45  52  000A0 P.AAO:  .ASCII  \RED\<0>                  ;
                                  010E0003  000A4 P.AAN:  .LONG   17694723                     ;
                                  00000000' 000A8         .ADDRESS P.AAO                       ;
                                00  00  44  45  000AC P.AAQ:  .ASCII  \ED\<0><0>               ;
                                  010E0002  000B0 P.AAP:  .LONG   17694722                     ;
                                  00000000' 000B4         .ADDRESS P.AAQ                       ;
                                00  44  57  52  000B8 P.AAS:  .ASCII  \RWD\<0>                 ;
                                  010E0003  000BC P.AAR:  .LONG   17694723                     ;
                                  00000000' 000C0         .ADDRESS P.AAS                       ;
                                00  00  44  57  000C4 P.AAU:  .ASCII  \WD\<0><0>               ;
                                  010E0002  000C8 P.AAT:  .LONG   17694722                     ;
                                  00000000' 000CC         .ADDRESS P.AAU                       ;
                                00  00  44  52  000D0 P.AAW:  .ASCII  \RD\<0><0>               ;
                                  010E0002  0C0D4 P.AAV:  .LONG   17694722                     ;
                                  00000000' 000D8         .ADDRESS P.AAW                       ;
                                00  00  00  44  000DC P.AAY:  .ASCII  \D\<0><0><0>             ;
                                  010E0001  000E0 P.AAX:  .LONG   17694721                     ;
                                  00000000' 000E4         .ADDRESS P.AAY                       ;
                                00  45  57  52  000E8 P.ABA:  .ASCII  \RWE\<0>                ;
                                  010E0003  000EC P.AAZ:  .LONG   17694723                     ;
                                  00000000' 000F0         .ADDRESS P.ABA                       ;
                                00  00  45  57  000F4 P.ABC:  .ASCII  \WE\<0><0>               ;
                                  010E0002  000F8 P.ABB:  .LONG   17694722                     ;
                                  00000000' 000FC         .ADDRESS P.ABC                       ;
                                00  00  45  52  00100 P.ABE:  .ASCII  \RE\<0><0>               ;
                                  010E0002  00104 P.ABD:  .LONG   17694722                     ;
                                  00000000' 00108         .ADDRESS P.ABE                       ;
```

```
00  00  00  45  0010C P.ABG:   .ASCII  \E\<0><0><0>
        010E0001  00110 P.ABF:   .LONG   17694721
        00000000' 00114          .ADDRESS P.ABG
00  00  57  52  00118 P.ABI:   .ASCII  \RW\<0><0>
        010E0002  0011C P.ABH:   .LONG   17694722
        00000000' 00120          .ADDRESS P.ABI
00  00  00  57  00124 P.ABK:   .ASCII  \W\<0><0><0>
        010E0001  00128 P.ABJ:   .LONG   17694721
        00000000' 0012C          .ADDRESS P.ABK
00  00  00  52  00130 P.ABM:   .ASCII  \R\<0><0><0>
        010E0001  00134 P.ABL:   .LONG   17694721
        00000000' 00138          .ADDRESS P.ABM
                  0013C P.ABO:   .BLKB   0
        010E0000  0013C P.ABN:   .LONG   17694720
        00000000' 00140          .ADDRESS P.ABO
50  4C  57  52  00144 P.ABQ:   .ASCII  \RWLP\
        010E0004  00148 P.ABP:   .LONG   17694724
        00000000' 0014C          .ADDRESS P.ABQ
00  50  4C  57  00150 P.ABS:   .ASCII  \WLP\<0>
        010E0003  00154 P.ABR:   .LONG   17694723
        00000000' 00158          .ADDRESS P.ABS
00  50  4C  52  0015C P.ABU:   .ASCII  \RLP\<0>
        010E0003  00160 P.ABT:   .LONG   17694723
        00000000' 00164          .ADDRESS P.ABU
00  00  50  4C  00168 P.ABW:   .ASCII  \LP\<0><0>
        010E0002  0016C P.ABV:   .LONG   17694722
        00000000' 00170          .ADDRESS P.ABW
00  4C  57  52  00174 P.ABY:   .ASCII  \RWL\<0>
        010E0003  00178 P.ABX:   .LONG   17694723
        00000000' 0017C          .ADDRESS P.ABY
00  00  4C  57  00180 P.ACA:   .ASCII  \WL\<0><0>
        010E0002  00184 P.ABZ:   .LONG   17694722
        00000000' 00188          .ADDRESS P.ACA
00  00  4C  52  0018C P.ACC:   .ASCII  \RL\<0><0>
        010E0002  00190 P.ACB:   .LONG   17694722
        00000000' 00194          .ADDRESS P.ACC
00  00  00  4C  00198 P.ACE:   .ASCII  \L\<0><0><0>
        010E0001  0019C P.ACD:   .LONG   17694721
        00000000' 001A0          .ADDRESS P.ACE
00  50  57  52  001A4 P.ACG:   .ASCII  \RWP\<0>
        010E0003  001A8 P.ACF:   .LONG   17694723
        00000000' 001AC          .ADDRESS P.ACG
00  00  50  57  001B0 P.ACI:   .ASCII  \WP\<0><0>
        010E0002  001B4 P.ACH:   .LONG   17694722
        00000000' 001B8          .ADDRESS P.ACI
00  00  50  52  001BC P.ACK:   .ASCII  \RP\<0><0>
        010E0002  001C0 P.ACJ:   .LONG   17694722
        00000000' 001C4          .ADDRESS P.ACK
00  00  00  50  001C8 P.ACM:   .ASCII  \P\<0><0><0>
        010E0001  001CC P.ACL:   .LONG   17694721
        00000000' 001D0          .ADDRESS P.ACM
00  00  57  52  001D4 P.ACO:   .ASCII  \RW\<0><0>
        010E0002  001D8 P.ACN:   .LONG   17694722
        00000000' 001DC          .ADDRESS P.ACO
00  00  00  57  001E0 P.ACQ:   .ASCII  \W\<0><0><0>
        010E0001  001E4 P.ACP:   .LONG   17694721
        00000000' 001E8          .ADDRESS P.ACQ
```

```
                                    00  00  00  52  001EC P.ACS:  .ASCII  \R\<0><0><0>                                  ;
                                          010E0001  001F0 P.ACR:  .LONG   17694721                                    ;
                                          00000000' 001F4         .ADDRESS P.ACS                                      ;
                                                    001F8 P.ACU:  .BLKB   0                                           ;
                                          010E0000  001F8 P.ACT:  .LONG   17694720                                    ;
                                          00000000' 001FC         .ADDRESS P.ACU                                      ;

                                                                  .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00014 RWED_VALUES:
                                                                  .ADDRESS P.AAJ, P.AAL, P.AAN, P.AAP, P.AAR, -       ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0002C         P.AAT, P.AAV, P.AAX, P.AAZ, P.ABB, P.ABD, - ;
                    00000000' 00000000' 00000000' 00000000' 00044         P.ABF, P.ABH, P.ABJ, P.ABL, P.ABN           ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00054 RWPL_VALUES:
                                                                  .ADDRESS P.ABP, P.ABR, P.ABT, P.ABV, P.ABX, -       ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0006C         P.ABZ, P.ACB, P.ACD, P.ACF, P.ACH, P.ACJ, - ;
                    00000000' 00000000' 00000000' 00000000' 00084         P.ACL, P.ACN, P.ACP, P.ACR, P.ACT           ;


                                                                  .PSECT  $CODE$,NOWRT,2

                                          0004 00000              .ENTRY  EXPAND_PROT, Save R2                        ; 0263
                                    0C    AC   D5 00002           TSTL    KEY                                         ; 0331
                                          1B   12 00005           BNEQ    2$                                          ;
                                          50   D4 00007           CLRL    INDEX                                       ; 0332
                52                  50         02   78 00009 1$:  ASHL    #2, INDEX, R2                               ; 0333
      51        08    AC                 04     52   EF 0000D     EXTZV   R2, #4, PROT, R1                            ;
                                 04 BC40 00000000'EF41   D0 00013 MOVL    RWED_VALUES[R1], @TABLE[INDEX]              ;
                E8                  50         03   F3 0001D     AOBLEQ  #3, INDEX, 1$                               ;
                                                04   00021       RET                                                 ; 0332
                                          50   D4 00022 2$:     CLRL    INDEX                                       ; 0334
                52                  50         02   78 00024 3$:  ASHL    #2, INDEX, R2                               ; 0335
      51        08    AC                 04     52   EF 00028     EXTZV   R2, #4, PROT, R1                            ;
                                 04 BC40 00000000'EF41   D0 0002E MOVL    RWPL_VALUES[R1], @TABLE[INDEX]              ;
                E8                  50         03   F3 00038     AOBLEQ  #3, INDEX, 3$                               ;
                                                04   0003C       RET                                                 ; 0338

; Routine Size:  61 bytes,    Routine Base:  $CODE$ + 010F
```

```
;  346          0339  1 END
;  347          0340  0 ELUDOM


                                                        .EXTRN  LIB$STOP

;                                PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  $OWN$                         148  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _LIB$KEY0$                      0  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  _LIB$STATE$                    10  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;  $PLIT$                        512  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        332  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                          Library Statistics

;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]STARLET.L32;1   9776        15         0         581        00:01.0
;  _$255$DUA28:[SYSLIB]TPAMAC.L32;1      42        19        45          14        00:00.1


;                                COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:UTILSUBS/OBJ=OBJ$:UTILSUBS MSRC$:UTILSUBS/UPDATE=(ENH$:UTILSUBS)

; Size:          332 code + 670 data bytes
; Run Time:          00:10.8
; Elapsed Time:      00:36.3
; Lines/CPU Min:      1883
; Lexemes/CPU-Min:  27063
; Memory Used:   100 pages
; Compilation Complete
```

TYPEMAIN
LIS

COBPROLOG
REQ

COBACCDAT
LIS

COBACCDWK
LIS

UNLOCK
LIS

UTILSUBS
LIS

INTPAR
SDL

COBDEF
REQ

COBACCDAY
LIS

COBACCECV
LIS

COBRTL

COBLNK
REQ

COBRTL
MAP