I 3

```
   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
```

```
**FILE**ID**SETACT                                        I  1





  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT    AAAAAA      CCCCCCCC  TTTTTTTTTT
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT    AAAAAA      CCCCCCCC  TTTTTTTTTT
SS          EE              TT      AA      AA  CC              TT
SS          EE              TT      AA      AA  CC              TT
SS          EE              TT      AA      AA  CC              TT
SS          EE              TT      AA      AA  CC              TT
  SSSSSS    EEEEEEEE        TT      AA      AA  CC              TT
  SSSSSS    EEEEEEEE        TT      AA      AA  CC              TT
        SS  EE              TT      AAAAAAAAAA  CC              TT
        SS  EE              TT      AAAAAAAAAA  CC              TT
        SS  EE              TT      AA      AA  CC              TT          ....
        SS  EE              TT      AA      AA  CC              TT          ....
SSSSSSSS    EEEEEEEEEE      TT      AA      AA    CCCCCCCC      TT          ....
SSSSSSSS    EEEEEEEEEE      TT      AA      AA    CCCCCCCC      TT          ....


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
```

```
0 MODULE setact   (
0                 IDENT = 'V04-000',
0                 ADDRESSING_MODE(EXTERNAL=GENERAL,
0                                 NONEXTERNAL=LONG_RELATIVE)
0                 ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:     Set
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains the action routines for SET FILE, SET DIRECTORY,
1 !       and SET VOLUME.
1 !
1 ! ENVIRONMENT:
1 !
1 !       Vax native, privileged user mode
1 !
1 !--
1
1 ! AUTHOR:       Gerry Smith                CREATION DATE: 04-Aug-1981
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-005 GAS0047         Gerry Smith             15-Feb-1982
1 !               Only get the file name for SET FILE/ENTER=filename.
1 !               The $PARSE is moved to SETFILE, so that stickiness
1 !               can be applied with the input file.
1 !
1 !       V03-004 GAS0038         Gerry Smith             2-Feb-1982
1 !               Add /GLOBAL_BUFFERS action routine for SET FILE.
```

```
:   1 !
:   1 !        V03-003 GAS0030         Gerry Smith             1-Jan-1982
:   1 !                Add /RETENTION action routine, for SET VOLUME.
:   1 !
:   1 !        V03-002 GAS0026         Gerry Smith            18-Dec-1981
:   1 !                Use shared message file, and lower fatal messages to
:   1 !                simple error messages.
:   1 !
:   1 !        V03-001 GAS0021         Gerry Smith            30-Nov-1981
:   1 !                Allow zero values for group and member of UIC
:   1 !
:   1 !**
```

```
;    71         0070  1 LIBRARY 'SYS$LIBRARY:LIB';
;    72         0071  1 LIBRARY 'SYS$LIBRARY:CLIMAC';
;    73         0072  1 LIBRARY 'SYS$LIBRARY:TPAMAC';
;    74         0073  1
;    75         0074  1 STRUCTURE
;    76         0075  1     BBLOCK [O, P, S, E; N] =
;    77         0076  1         [N]
;    78         0077  1         (BBLOCK + O)<P,S,E>;
```

```
1 FORWARD ROUTINE
1                                       ! Action routines for:
1       acc_act,                        ! /ACCESSED             (VOLUME)
1       back_act,                       ! /BACKUP               (FILE)
1       noback_act,                     ! /NOBACKUP             (FILE)
1       data_act,                       ! /DATA_CHECK           (VOLUME,FILE)
1       enter_act,                      ! /ENTER                (FILE)
1       erase_act,                      ! /ERASE_ON_DELETE      (FILE)
1       noerase_act,                    ! /NOERASE_ON_DELETE    (FILE)
1       exp_act,                        ! /EXPIRATION_DATE      (FILE)
1       noexp_act,                      ! /NOEXPIRATION_DATE    (FILE)
1       ext_act,                        ! /EXTENSION            (FILE,VOLUME)
1       fprot_act,                      ! /FILE_PROTECTION      (VOLUME)
1       gbuf_act,                       ! /GLOBAL_BUFFERS       (FILE)
1       journal_act,                    ! /JOURNAL              (FILE)
1       label_act,                      ! /LABEL                (VOLUME)
1       owner_act,                      ! /OWNER_UIC            (ALL)
1       retent_act,                     ! /RETENTION            (VOLUME)
1       test_char,                      ! action routine used by retent_act
1       user_act,                       ! /USER_NAME            (VOLUME)
1       vprot_act,                      ! /PROTECTION           (VOLUME)
1       vrsn_act,                       ! /VERSION_LIMIT        (DIRECTORY,FILE)
1       window_act;                     ! /WINDOWS              (VOLUME)
1
1 EXTERNAL ROUTINE
1       calculate_max,
1       sys$fao,
1       lib$tparse,
1       lib$cvt_time,
1       lib$cvt_dtime,
1       lib$cvt_dtb;
1
1 !
1 ! External data references
1 !
1 EXTERNAL
1       rename_buf : VECTOR[nam$c_maxrss,BYTE],      ! Name buffer for /ENTER
1       file_name : VECTOR[2],                       ! File name descriptor
1       file_rlf : BBLOCK[nam$c_bln],                ! Related name block
1
1       set$l_status,                   ! Status return for SET dispatcher
1       set$a_cliwork;                  ! CLI work area in SET dispatcher
1
1 !
1 ! Literal data definitions
1 !
1 LITERAL
1       true = 1,
1       false = 0;
1
```

```
;   131         0128  1  !
;   132         0129  1  ! Define the qualifier flag bits used by all SET FILE/DIRECTORY/VOLUME
;   133         0130  1  !
;   134         0131  1  GLOBAL LITERAL
;   135       P 0132  1      $EQULST
;   136       P 0133  1          (QUAL_,,1,1,
;   137       P 0134  1          (access,),              ! ACCESSED              (VOLUME)
;   138       P 0135  1          (backup,),              ! BACKUP                (FILE)
;   139       P 0136  1          (nobackup,),            ! NOBACKUP              (FILE)
;   140       P 0137  1          (confirm,),             ! CONFIRM               (ALL)
;   141       P 0138  1          (data,),                ! DATA_CHECK            (FILE, VOLUME)
;   142       P 0139  1          (enter,),               ! ENTER                 (FILE)
;   143       P 0140  1          (eof,),                 ! END_OF_FILE           (FILE)
;   144       P 0141  1          (erase,),               ! ERASE_ON_DELETE       (FILE)
;   145       P 0142  1          (noerase,),             ! NOERASE_ON_DELETE     (FILE)
;   146       P 0143  1          (expi,),                ! EXPIRATION_DATE       (FILE)
;   147       P 0144  1          (exte,),                ! EXTENSION             (FILE, VOLUME)
;   148       P 0145  1          (fprot,),               ! FILE_PROTECTION       (VOLUME)
;   149       P 0146  1          (gbuf,),                        ! GLOBAL_BUFFERS        (FILE)
;   150       P 0147  1          (journal,),             ! JOURNAL               (FILE)
;   151       P 0148  1          (label,),               ! LABEL                 (VOLUME)
;   152       P 0149  1          (log,),                 ! LOG                   (ALL)
;   153       P 0150  1          (nodi,),                ! NODIRECTORY           (FILE)
;   154       P 0151  1          (owner,),               ! OWNER_UIC             (ALL)
;   155       P 0152  1          (parent,),              ! OWNER=PARENT          (FILE)
;   156       P 0153  1          (remove,),              ! REMOVE                (FILE)
;   157       P 0154  1          (retent,),              ! RETENTION             (VOLUME)
;   158       P 0155  1          (rprot,),               ! RECORD_PROTECTION     (FILE, VOLUME)
;   159       P 0156  1          (trunc,),               ! TRUNCATE              (FILE)
;   160       P 0157  1          (username,),            ! USERNAME              (VOLUME)
;   161       P 0158  1          (vprot,),               ! PROTECTION            (VOLUME)
;   162       P 0159  1          (vrsn,),                ! VERSION_LIMIT         (FILE, DIRECTORY)
;   163         0160  1          (windows,));            ! WINDOWS               (VOLUME)
;   164         0161  1
;   165         0162  1  !
;   166         0163  1  ! Define the DATA_CHECK option bits
;   167         0164  1  !
;   168         0165  1  GLOBAL LITERAL
;   169       P 0166  1      $EQULST
;   170       P 0167  1          (DATA_,,1,1,
;   171       P 0168  1          (read,),
;   172       P 0169  1          (noread,),
;   173       P 0170  1          (write,),
;   174         0171  1          (nowrite,));
;   175         0172  1  !
;   176         0173  1  ! Define the JOURNAL option bits
;   177         0174  1  !
;   178         0175  1  GLOBAL LITERAL
;   179       P 0176  1      $EQULST
;   180       P 0177  1          (JRNL_,,1,1,
;   181       P 0178  1          (ai,),
;   182       P 0179  1          (noai,),
;   183       P 0180  1          (at,),
;   184       P 0181  1          (noat,),
;   185       P 0182  1          (bi,),
;   186       P 0183  1          (nobi,),
;   187       P 0184  1          (ru,),
;  188        P 0185  1          (noru,),
;  189        P 0186  1          (rum,),
;  190          0187  1          (norum,));
;  191          0188  1
;  192          0189  1
```

```
1 !
1 ! Declare external references
1 !
1 EXTERNAL
1     setfile$flags : BITVECTOR[32],                      ! Qualifier flags
1     setfile$dflags : BITVECTOR[32],                     ! Data_check flag word
1     setfile$jflags : BITVECTOR[32],                     ! Journal flag word
1     acc_value,                                          ! LRU value
1     exp_value : BBLOCK[8],                              ! Expiration date
1     exte_value,                                         ! Extension quantity
1     fprot_value,                                        ! File_protection value
1     gbuf_value,                                         ! Global buffer count
1     label_value : VECTOR[2],                            ! Volume label descriptor
1     uic_value,                                          ! Owner uic
1     group,                                              ! Group number
1     member,                                             ! Member number
1     user_value : VECTOR[2],                             ! Username string descriptor
1     retmin_value : VECTOR[2],                           ! Minimum retention time quadword
1     retmax_value : VECTOR[2],                           ! Maximum retention time quadword
1     vprot_value,                                        ! Volume protection value
1     vrsn_value,                                         ! Version limit
1     window_value;                                       ! Window length
1
1
1 !
1 ! Declare the error messages
1 !
1 EXTERNAL LITERAL
1     set$_facility,                          ! SET facility code
1     set$_operreq,                           ! OPER privilege required
1     set$_writeerr;                          ! Error accessing file
```

```
;   226          0221   1 !
;   227          0222   1 ! Define the TPARSE block
;   228          0223   1 !
;   229          0224   1 OWN
;   230          0225   1     tparse_block : BBLOCK[tpa$k_length0]        ! TPARSE block
;   231          0226   1                INITIAL(tpa$k_count0,
;   232          0227   1                        tpa$m_blanks OR
;   233          0228   1                        tpa$m_abbrev);
;   234          0229   1
;  236            0230  1 !
;  237            0231  1 ! TPARSE table for /DATA_CHECK options
;  238            0232  1 !
;  239            0233  1 $INIT_STATE (dc_state,dc_keys);
;  240            0234  1
;  241          P 0235  1 $STATE  (optstart,
;  242          P 0236  1         (tpa$_eos,tpa$_exit,,data_write,setfile$dflags),! default is WRITE
;  243            0237  1         (tpa$_lambda,getoption));                     ! fall thru to options
;  244            0238  1
;  245          P 0239  1 $STATE  (getoption,
;  246          P 0240  1         ('READ',,,1^data_read,setfile$dflags),        ! READ present
;  247          P 0241  1         ('WRITE',,,1^data_write,setfile$dflags),      ! WRITE present
;  248          P 0242  1         ('NOREAD',,,1^data_noread,setfile$dflags),    ! NOREAD present
;  249            0243  1         ('NOWRITE',,,1^data_nowrite,setfile$dflags)); ! NOWRITE present
;  250            0244  1
;  251          P 0245  1 $STATE  (,
;  252          P 0246  1         (tpa$_eos,tpa$_exit),                         ! either end of line
;  253            0247  1         (',',getoption));                             ! or get rid of the comma
;  254            0248  1
;   256            0249   1 !
;   257            0250   1 ! TPARSE table for /OWNER_UIC option
;   258            0251   1 !
;   259            0252   1 $INIT_STATE (owner_state,owner_keys);
;   260            0253   1
;   261          P 0254   1 $STATE  (ownerstart,
;   262          P 0255   1         ('PARENT',tpa$_exit,,1^qual_parent,setfile$flags),      ! Check for PARENT
;   263          P 0256   1         ('['),                                         ! Look for square bracket
;   264            0257   1         ('<'));                                        ! Or a squiggle
;   265            0258   1
;   266          P 0259   1 $STATE  (,
;   267            0260   1         (tpa$_octal,,,,group));                        ! Get group number
;   268            0261   1
;   269          P 0262   1 $STATE  (,
;   270            0263   1         (','));                                        ! Get rid of the comma
;   271            0264   1
;   272          P 0265   1 $STATE  (,
;   273            0266   1         (tpa$_octal,,,,member));                       ! Get member number
;   274            0267   1
;   275          P 0268   1 $STATE  (,
;   276          P 0269   1         (']'),                                         ! Get end bracket
;   277            0270   1         ('>'));
;   278            0271   1
;   279          P 0272   1 $STATE  (,
;   280            0273   1         (tpa$_eos,tpa$_exit));                         ! Clean-up
;   282          0274  1 !
;   283          0275  1 ! TPARSE table for /JOURNAL option
;   284          0276  1 !
;   285          0277  1 $INIT_STATE (journal_state, journal_keys);
;   286          0278  1
;   287        P 0279  1 $STATE  (getjopts,
;   288        P 0280  1          ('AI',,,1^jrnl_ai,setfile$jflags),       ! AI journaling
;   289        P 0281  1          ('AT',,,1^jrnl_at,setfile$jflags),       ! AT journaling
;   290        P 0282  1          ('BI',,,1^jrnl_bi,setfile$jflags),       ! BI journaling
;   291        P 0283  1          ('NOAI',,,1^jrnl_noai,setfile$jflags),   ! No AI journaling
;   292        P 0284  1          ('NOAT',,,1^jrnl_noat,setfile$jflags),   ! No AT journaling
;   293        P 0285  1          ('NOBI',,,1^jrnl_nobi,setfile$jflags),   ! No BI journaling
;   294        P 0286  1          ('NORU',,,1^jrnl_noru,setfile$jflags),   ! No RU journaling
;   295        P 0287  1          ('NORUM',,,1^jrnl_norum,setfile$jflags),! No RUM journaling
;   296        P 0288  1          ('RU',,,1^jrnl_ru,setfile$jflags),       ! RU journaling
;   297          0289  1          ('RUM',,,1^jrnl_rum,setfile$jflags));    ! RUM journaling
;   298          0290  1
;   299        P 0291  1 $STATE  (,
;   300        P 0292  1          (tpa$_eos,tpa$_exit),                    ! Either the end
;   301          0293  1          (',',getjopts));                         ! Or more to come
```

```
!
! TPARSE table for protection
!

$INIT_STATE (pro_state, pro_keys);

$STATE  (NEXTPRO,
        ('SYSTEM', SYPR,, %X'000F0000', fprot_value),
        ('OWNER',  OWPR,, %X'00F00000', fprot_value),
        ('GROUP',  GRPR,, %X'0F000000', fprot_value),
        ('WORLD',  WOPR,, %X'F0000000', fprot_value)
        );

$STATE  (SYPR,
        (':'),
        ('='),
        (TPA$_LAMBDA, ENDPRO)
        );

$STATE  (SYPRO,
        ('R', SYPRO,, %X'0001', fprot_value),
        ('W', SYPRO,, %X'0002', fprot_value),
        ('E', SYPRO,, %X'0004', fprot_value),
        ('P', SYPRO,, %X'0004', fprot_value),
        ('D', SYPRO,, %X'0008', fprot_value),
        ('L', SYPRO,, %X'0008', fprot_value),
        (TPA$_LAMBDA, ENDPRO)
        );

$STATE  (OWPR,
        (':'),
        ('='),
        (TPA$_LAMBDA, ENDPRO)
        );

$STATE  (OWPRO,
        ('R', OWPRO,, %X'0010', fprot_value),
        ('W', OWPRO,, %X'0020', fprot_value),
        ('E', OWPRO,, %X'0040', fprot_value),
        ('P', OWPRO,, %X'0040', fprot_value),
        ('D', OWPRO,, %X'0080', fprot_value),
        ('L', OWPRO,, %X'0080', fprot_value),
        (TPA$_LAMBDA, ENDPRO)
        );

$STATE  (GRPR,
        (':'),
        ('='),
        (TPA$_LAMBDA, ENDPRO)
        );

$STATE  (GRPRO,
        ('R', GRPRO,, %X'0100', fprot_value),
        ('W', GRPRO,, %X'0200', fprot_value),
        ('E', GRPRO,, %X'0400', fprot_value),
        ('P', GRPRO,, %X'0400', fprot_value),
        ('D', GRPRO,, %X'0800', fprot_value),
                ('L', GRPRO,, %X'0800', fprot_value),
                (TPA$_LAMBDA, ENDPRO)
                );

        $STATE  (WOPR,
                (':'),
                ('='),
                (TPA$_LAMBDA, ENDPRO)
                );

        $STATE  (WOPRO,
                ('R', WOPRO,, %X'1000', fprot_value),
                ('W', WOPRO,, %X'2000', fprot_value),
                ('E', WOPRO,, %X'4000', fprot_value),
                ('P', WOPRO,, %X'4000', fprot_value),
                ('D', WOPRO,, %X'8000', fprot_value),
                ('L', WOPRO,, %X'8000', fprot_value),
                (TPA$_LAMBDA, ENDPRO)
                );

        $STATE  (ENDPRO,
                (',', NEXTPRO),
                (TPA$_EOS, TPA$_EXIT)
                );
```

```
;  385          0375  1 !
;  386          0376  1 ! Tparse table for /RETENTION option
;  387          0377  1 !
;  388          0378  1 $INIT_STATE (ret_state, ret_keys)
;  389          0379  1
;  390        P 0380  1 $STATE  (retstart,
;  391          0381  1         ((get_delta),,,,retmin_value));          ! Get the first delta string
;  392          0382  1
;  393        P 0383  1 $STATE  (,
;  394        P 0384  1         (','),                                   ! If a comma, get next string
;  395          0385  1         (tpa$_eos,tpa$_exit));                   ! Else exit
;  396          0386  1
;  397        P 0387  1 $STATE  (,
;  398          0388  1         ((get_delta),,,,retmax_value));          ! Get the next delta string
;  399          0389  1
;  400        P 0390  1 $STATE  (,
;  401          0391  1         (tpa$_eos,tpa$_exit));                   ! And exit
;  402          0392  1
;  403        P 0393  1 $STATE  (get_delta,
;  404        P 0394  1         (tpa$_any,get_delta,test_char),          ! Get next character in delta string
;  405          0395  1         (tpa$_lambda,tpa$_exit));
```

```
  1 GLOBAL ROUTINE acc_act (option_block, callback) =
  1 !++
  1 !
  1 ! This is the action routine for the /ACCESSED qualifier.  It first checks to
  1 ! make sure that the process has OPER privilege.  If so, then the ACCESS value
  1 ! is obtained and bounds checking is performed on it.
  1 !
  1 !--
  2 BEGIN
  2
  2 OWN privs : BBLOCK[8];              ! Place to store the process privileges
  2
  2 LOCAL
  2     status,                         ! Status return
  2     desc : BBLOCK[dsc$c_s_bln];     ! General descriptor
  2
  2 MAP option_block : REF BBLOCK;  ! Define the CLI block
  2
  2 !
  2 ! Call $SETPRV to get the current privileges of the process.  If the process
  2 ! does not have OPER, then signal an error and stop.
  2 !
P 3 IF NOT (status = $SETPRV(ENBFLG = 1,               ! Enable
P 3                          PRVADR = 0,               ! No new privileges
P 3                          PRMFLG = 1,               ! Get current privileges
  3                          PRVPRV = privs))
  2 THEN SIGNAL_STOP(.status);
  2
  2 IF NOT .privs[prv$v_oper] THEN SIGNAL_STOP(set$_operreq);
  2
  2 !
  2 ! The process has the correct privilege, so go ahead and get the value
  2 !
  2
  2 acc_value = 3;                      ! Set up the default
  2
  2 !
  2 ! If a value was specified, use it; otherwise, use the default.
  2 !
  2 IF .option_block[cli$w_qdvalsiz] EQL 0
  2 THEN RETURN true;
  2
  2 !
  2 ! Convert the value
  2 !
  3 IF NOT (status = LIB$CVT_DTB(.option_block[cli$w_qdvalsiz],
  3                              .option_block[cli$a_qdvaladr],
  3                              acc_value))
  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,  ! Signal a syntax error
  2                 1,
  2                 option_block[cli$q_qdvaldesc],
  2                 .status)
  2 ELSE
  3     BEGIN
  4     IF NOT (.acc_value GEQ 0            ! Check that value is in range
  4                 AND
  4             .acc_value LEQ 255)
```

```
;  464          0453  3       THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,      ! If not, exit with an error.
;  465          0454  3                        1,
;  466          0455  3                        option_block[cli$q_qdvaldesc],
;  467          0456  3                        set$_facility^16 + shr$_valerr + sts$k_error);
;  468          0457  2       END;
;  469          0458  2    RETURN true;
;  470          0459  1    END;


                                                         .TITLE  SETACT
                                                         .IDENT  \V04-000\

                                                         .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                         00000  ;TPA$KEYST0
                                                U.9:     .BLKB   0
                             44  41  45  52  00000  ;TPA$KEYST
                                                U.11:    .ASCII  \READ\                                      ;
                                         FF  00004       .BYTE   -1                                          ;
                                         00005  ;TPA$KEYST0
                                                U.15:    .BLKB   0
                         45  54  49  52  57  00005  ;TPA$KEYST
                                                U.17:    .ASCII  \WRITE\                                     ;
                                         FF  0000A       .BYTE   -1                                          ;
                                         0000B  ;TPA$KEYST0
                                                U.21:    .BLKB   0
                     44  41  45  52  4F  4E  0000B  ;TPA$KEYST
                                                U.23:    .ASCII  \NOREAD\                                    ;
                                         FF  00011       .BYTE   -1                                          ;
                                         00012  ;TPA$KEYST0
                                                U.27:    .BLKB   0
                 45  54  49  52  57  4F  4E  00012  ;TPA$KEYST
                                                U.29:    .ASCII  \NOWRITE\                                   ;
                                         FF  00019       .BYTE   -1                                          ;
                                         FF  0001A  ;TPA$KEYFILL
                                                U.33:    .BYTE   -1                                          ;
                                         0001B  ;TPA$KEYST0
                                                U.39:    .BLKB   0
                     54  4E  45  52  41  50  0001B  ;TPA$KEYST
                                                U.41:    .ASCII  \PARENT\                                    ;
                                         FF  00021       .BYTE   -1                                          ;
                                         FF  00022  ;TPA$KEYFILL
                                                U.48:    .BYTE   -1                                          ;
                                         00023  ;TPA$KEYST0
                                                U.59:    .BLKB   0
                                     49  41  00023  ;TPA$KEYST
                                                U.61:    .ASCII  \AI\                                        ;
                                         FF  00025       .BYTE   -1                                          ;
                                         00026  ;TPA$KEYST0
                                                U.65:    .BLKB   0
                                     54  41  00026  ;TPA$KEYST
                                                U.67:    .ASCII  \AT\                                        ;
                                         FF  00028       .BYTE   -1                                          ;
                                         00029  ;TPA$KEYST0
                                                U.71:    .BLKB   0
                                     49  42  00029  ;TPA$KEYST
                                                U.73:    .ASCII  \BI\                                        ;
                         FF  0002B          .BYTE   -1                                 ;
                             0002C ;TPA$KEYST0
                                   U.77:    .BLKB   0
             49  41  4F  4E  0002C ;TPA$KEYST
                                   U.79:    .ASCII  \NOAI\                             ;
                         FF  00030          .BYTE   -1                                 ;
                             00031 ;TPA$KEYST0
                                   U.83:    .BLKB   0
             54  41  4F  4E  00031 ;TPA$KEYST
                                   U.85:    .ASCII  \NOAT\                             ;
                         FF  00035          .BYTE   -1                                 ;
                             00036 ;TPA$KEYST0
                                   U.89:    .BLKB   0
             49  42  4F  4E  00036 ;TPA$KEYST
                                   U.91:    .ASCII  \NOBI\                             ;
                         FF  0003A          .BYTE   -1                                 ;
                             0003B ;TPA$KEYST0
                                   U.95:    .BLKB   0
             55  52  4F  4E  0003B ;TPA$KEYST
                                   U.97:    .ASCII  \NORU\                             ;
                         FF  0003F          .BYTE   -1                                 ;
                             00040 ;TPA$KEYST0
                                   U.101:   .BLKB   0
         4D  55  52  4F  4E  00040 ;TPA$KEYST
                                   U.103:   .ASCII  \NORUM\                            ;
                         FF  00045          .BYTE   -1                                 ;
                             00046 ;TPA$KEYST0
                                   U.107:   .BLKB   0
                     55  52  00046 ;TPA$KEYST
                                   U.109:   .ASCII  \RU\                               ;
                         FF  00048          .BYTE   -1                                 ;
                             00049 ;TPA$KEYST0
                                   U.113:   .BLKB   0
                 4D  55  52  00049 ;TPA$KEYST
                                   U.115:   .ASCII  \RUM\                              ;
                         FF  0004C          .BYTE   -1                                 ;
                         FF  0004D ;TPA$KEYFILL
                                   U.119:   .BYTE   -1                                 ;
                             0004E ;TPA$KEYST0
                                   U.125:   .BLKB   0
     4D  45  54  53  59  53  0004E ;TPA$KEYST
                                   U.127:   .ASCII  \SYSTEM\                           ;
                         FF  00054          .BYTE   -1                                 ;
                             00055 ;TPA$KEYST0
                                   U.133:   .BLKB   0
         52  45  4E  57  4F  00055 ;TPA$KEYST
                                   U.135:   .ASCII  \OWNER\                            ;
                         FF  0005A          .BYTE   -1                                 ;
                             0005B ;TPA$KEYST0
                                   U.141:   .BLKB   0
         50  55  4F  52  47  0005B ;TPA$KEYST
                                   U.143:   .ASCII  \GROUP\                            ;
                         FF  00060          .BYTE   -1                                 ;
                             00061 ;TPA$KEYST0
                                   U.149:   .BLKB   0
         44  4C  52  4F  57  00061 ;TPA$KEYST
                                   U.151:   .ASCII  \WORLD\                            ;
```

```
                   FF  00066            .BYTE    -1                                  ;
                   FF  00067 ;TPA$KEYFILL
                             U.157:     .BYTE    -1                                  ;

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

                       00000 DC_STATE::
                                        .BLKB    0
                       00000 OPTSTART:
                                        .BLKB    0
                 71F7  00000 ;TPA$TYPE
                             U.2:       .WORD    29175                               ;
            00000000*  00002 ;TPA$ADDR
                             U.3:       .LONG    <<SETFILE$DFLAGS-U.3>-4>            ;
            00000003   00006 ;TPA$MASK
                             U.4:       .LONG    3                                   ;
                 FFFF  0000A ;TPA$TARGET
                             U.5:       .WORD    -1                                  ;
                 15F6  0000C ;TPA$TYPE
                             U.6:       .WORD    5622                                ;
                0000*  0000E ;TPA$TARGET
                             U.8:       .WORD    <<U.7-U.8>-2>                       ;
                       00010 ;GETOPTION
                             U.7:       .BLKB    0
                 6100  00010 ;TPA$TYPE
                             U.12:      .WORD    24832                               ;
            00000000*  00012 ;TPA$ADDR
                             U.13:      .LONG    <<SETFILE$DFLAGS-U.13>-4>           ;
            00000002   00016 ;TPA$MASK
                             U.14:      .LONG    2                                   ;
                 6101  0001A ;TPA$TYPE
                             U.18:      .WORD    24833                               ;
            00000000*  0001C ;TPA$ADDR
                             U.19:      .LONG    <<SETFILE$DFLAGS-U.19>-4>           ;
            00000008   00020 ;TPA$MASK
                             U.20:      .LONG    8                                   ;
                 6102  00024 ;TPA$TYPE
                             U.24:      .WORD    24834                               ;
            00000000*  00026 ;TPA$ADDR
                             U.25:      .LONG    <<SETFILE$DFLAGS-U.25>-4>           ;
            00000004   0002A ;TPA$MASK
                             U.26:      .LONG    4                                   ;
                 6503  0002E ;TPA$TYPE
                             U.30:      .WORD    25859                               ;
            00000000*  00030 ;TPA$ADDR
                             U.31:      .LONG    <<SETFILE$DFLAGS-U.31>-4>           ;
            00000010   00034 ;TPA$MASK
                             U.32:      .LONG    16                                  ;
                 11F7  00038 ;TPA$TYPE
                             U.34:      .WORD    4599                                ;
                 FFFF  0003A ;TPA$TARGET
                             U.35:      .WORD    -1                                  ;
                 142C  0003C ;TPA$TYPE
                             U.36:      .WORD    5164                                ;
                0000*  0003E ;TPA$TARGET
                             U.37:      .WORD    <<U.7-U.37>-2>                      ;
                       00040 OWNER_STATE::
                           .BLKB   0
           00040 OWNERSTART:
                           .BLKB   0
     7100  00040 ;TPA$TYPE
                 U.42:     .WORD   28928                                  ;
00000000*  00042 ;TPA$ADDR
                 U.43:     .LONG   <<SETFILE$FLAGS-U.43>-4>               ;
00080000   00046 ;TPA$MASK
                 U.44:     .LONG   524288                                 ;
     FFFF  0004A ;TPA$TARGET
                 U.45:     .WORD   -1                                     ;
     005B  0004C ;TPA$TYPE
                 U.46:     .WORD   91                                     ;
     043C  0004E ;TPA$TYPE
                 U.47:     .WORD   1084                                   ;
     45F4  00050 ;TPA$TYPE
                 U.49:     .WORD   17908                                  ;
00000000*  00052 ;TPA$ADDR
                 U.50:     .LONG   <<GROUP-U.50>-4>                       ;
     042C  00056 ;TPA$TYPE
                 U.51:     .WORD   1068                                   ;
     45F4  00058 ;TPA$TYPE
                 U.52:     .WORD   17908                                  ;
00000000*  0005A ;TPA$ADDR
                 U.53:     .LONG   <<MEMBER-U.53>-4>                      ;
     005D  0005E ;TPA$TYPE
                 U.54:     .WORD   93                                     ;
     043E  00060 ;TPA$TYPE
                 U.55:     .WORD   1086                                   ;
     15F7  00062 ;TPA$TYPE
                 U.56:     .WORD   5623                                   ;
     FFFF  00064 ;TPA$TARGET
                 U.57:     .WORD   -1                                     ;
           00066           .BLKB   2
           00068 JOURNAL_STATE::
                           .BLKB   0
           00068 GETJOPTS:
                           .BLKB   0
     6100  00068 ;TPA$TYPE
                 U.62:     .WORD   24832                                  ;
00000000*  0006A ;TPA$ADDR
                 U.63:     .LONG   <<SETFILE$JFLAGS-U.63>-4>              ;
00000002   0006E ;TPA$MASK
                 U.64:     .LONG   2                                      ;
     6101  00072 ;TPA$TYPE
                 U.68:     .WORD   24833                                  ;
00000000*  00074 ;TPA$ADDR
                 U.69:     .LONG   <<SETFILE$JFLAGS-U.69>-4>              ;
00000008   00078 ;TPA$MASK
                 U.70:     .LONG   8                                      ;
     6102  0007C ;TPA$TYPE
                 U.74:     .WORD   24834                                  ;
00000000*  0007E ;TPA$ADDR
                 U.75:     .LONG   <<SETFILE$JFLAGS-U.75>-4>              ;
00000020   00082 ;TPA$MASK
                 U.76:     .LONG   32                                     ;
     6103  00086 ;TPA$TYPE
```

```
                    U.80:    .WORD    24835                          ;
00000000*  00088  ;TPA$ADDR
                    U.81:    .LONG    <<SETFILE$JFLAGS-U.81>-4>      ;
00000004   0008C  ;TPA$MASK
                    U.82:    .LONG    4                              ;
    6104   00090  ;TPA$TYPE
                    U.86:    .WORD    24836                          ;
00000000*  00092  ;TPA$ADDR
                    U.87:    .LONG    <<SETFILE$JFLAGS-U.87>-4>      ;
00000010   00096  ;TPA$MASK
                    U.88:    .LONG    16                             ;
    6105   0009A  ;TPA$TYPE
                    U.92:    .WORD    24837                          ;
00000000*  0009C  ;TPA$ADDR
                    U.93:    .LONG    <<SETFILE$JFLAGS-U.93>-4>      ;
00000040   000A0  ;TPA$MASK
                    U.94:    .LONG    64                             ;
    6106   000A4  ;TPA$TYPE
                    U.98:    .WORD    24838                          ;
00000000*  000A6  ;TPA$ADDR
                    U.99:    .LONG    <<SETFILE$JFLAGS-U.99>-4>      ;
00000100   000AA  ;TPA$MASK
                    U.100:   .LONG    256                            ;
    6107   000AE  ;TPA$TYPE
                    U.104:   .WORD    24839                          ;
00000000*  000B0  ;TPA$ADDR
                    U.105:   .LONG    <<SETFILE$JFLAGS-U.105>-4>     ;
00000400   000B4  ;TPA$MASK
                    U.106:   .LONG    1024                           ;
    6108   000B8  ;TPA$TYPE
                    U.110:   .WORD    24840                          ;
00000000*  000BA  ;TPA$ADDR
                    U.111:   .LONG    <<SETFILE$JFLAGS-U.111>-4>     ;
00000080   000BE  ;TPA$MASK
                    U.112:   .LONG    128                            ;
    6509   000C2  ;TPA$TYPE
                    U.116:   .WORD    25865                          ;
00000000*  000C4  ;TPA$ADDR
                    U.117:   .LONG    <<SETFILE$JFLAGS-U.117>-4>     ;
00000200   000C8  ;TPA$MASK
                    U.118:   .LONG    512                            ;
    11F7   000CC  ;TPA$TYPE
                    U.120:   .WORD    4599                           ;
    FFFF   000CE  ;TPA$TARGET
                    U.121:   .WORD    -1                             ;
    142C   000D0  ;TPA$TYPE
                    U.122:   .WORD    5164                           ;
   0000*   000D2  ;TPA$TARGET
                    U.123:   .WORD    <<GETJOPTS-U.123>-2>           ;
           000D4  PRO_STATE::
                             .BLKB    0
           000D4  NEXTPRO:.BLKB    0
    7100   000D4  ;TPA$TYPE
                    U.128:   .WORD    28928                          ;
00000000*  000D6  ;TPA$ADDR
                    U.129:   .LONG    <<FPROT_VALUE-U.129>-4>        ;
000F0000   000DA  ;TPA$MASK
```

```
                                U.130:  .LONG   983040                                   ;
              0000* 000DE ;TPA$TARGET
                                U.132:  .WORD   <<U.131-U.132>-2>                        ;
              7101  000E0 ;TPA$TYPE
                                U.136:  .WORD   28929                                    ;
          00000000* 000E2 ;TPA$ADDR
                                U.137:  .LONG   <<FPROT_VALUE-U.137>-4>                  ;
          00F00000  000E6 ;TPA$MASK
                                U.138:  .LONG   15728640                                 ;
              0000* 000EA ;TPA$TARGET
                                U.140:  .WORD   <<U.139-U.140>-2>                        ;
              7102  000EC ;TPA$TYPE
                                U.144:  .WORD   28930                                    ;
          00000000* 000EE ;TPA$ADDR
                                U.145:  .LONG   <<FPROT_VALUE-U.145>-4>                  ;
          0F000000  000F2 ;TPA$MASK
                                U.146:  .LONG   251658240                                ;
              0000* 000F6 ;TPA$TARGET
                                U.148:  .WORD   <<U.147-U.148>-2>                        ;
              7503  000F8 ;TPA$TYPE
                                U.152:  .WORD   29955                                    ;
          00000000* 000FA ;TPA$ADDR
                                U.153:  .LONG   <<FPROT_VALUE-U.153>-4>                  ;
          F0000000  000FE ;TPA$MASK
                                U.154:  .LONG   -268435456                               ;
              0000* 00102 ;TPA$TARGET
                                U.156:  .WORD   <<U.155-U.156>-2>                        ;
                    00104 ;SYPR
                                U.131:  .BLKB   0
              003A  00104 ;TPA$TYPE
                                U.158:  .WORD   58                                       ;
              003D  00106 ;TPA$TYPE
                                U.159:  .WORD   61                                       ;
              15F6  00108 ;TPA$TYPE
                                U.160:  .WORD   5622                                     ;
              0000* 0010A ;TPA$TARGET
                                U.162:  .WORD   <<U.161-U.162>-2>                        ;
                    0010C SYPRO:    .BLKB   0
              7052  0010C ;TPA$TYPE
                                U.163:  .WORD   28754                                    ;
          00000000* 0010E ;TPA$ADDR
                                U.164:  .LONG   <<FPROT_VALUE-U.164>-4>                  ;
          00000001  00112 ;TPA$MASK
                                U.165:  .LONG   1                                        ;
              0000* 00116 ;TPA$TARGET
                                U.166:  .WORD   <<SYPRO-U.166>-2>                        ;
              7057  00118 ;TPA$TYPE
                                U.167:  .WORD   28759                                    ;
          00000000* 0011A ;TPA$ADDR
                                U.168:  .LONG   <<FPROT_VALUE-U.168>-4>                  ;
          00000002  0011E ;TPA$MASK
                                U.169:  .LONG   2                                        ;
              0000* 00122 ;TPA$TARGET
                                U.170:  .WORD   <<SYPRO-U.170>-2>                        ;
              7045  00124 ;TPA$TYPE
                                U.171:  .WORD   28741                                    ;
          00000000* 00126 ;TPA$ADDR
```

```
                            U.172:  .LONG   <<FPROT_VALUE-U.172>-4>             ;
         00000004  0012A ;TPA$MASK
                            U.173:  .LONG   4                                   ;
             0000*  0012E ;TPA$TARGET
                            U.174:  .WORD   <<SYPRO-U.174>-2>                   ;
             7050  00130 ;TPA$TYPE
                            U.175:  .WORD   28752                               ;
         00000000*  00132 ;TPA$ADDR
                            U.176:  .LONG   <<FPROT_VALUE-U.176>-4>             ;
         00000004  00136 ;TPA$MASK
                            U.177:  .LONG   4                                   ;
             0000*  0013A ;TPA$TARGET
                            U.178:  .WORD   <<SYPRO-U.178>-2>                   ;
             7044  0013C ;TPA$TYPE
                            U.179:  .WORD   28740                               ;
         00000000*  0013E ;TPA$ADDR
                            U.180:  .LONG   <<FPROT_VALUE-U.180>-4>             ;
         00000008  00142 ;TPA$MASK
                            U.181:  .LONG   8                                   ;
             0000*  00146 ;TPA$TARGET
                            U.182:  .WORD   <<SYPRO-U.182>-2>                   ;
             704C  00148 ;TPA$TYPE
                            U.183:  .WORD   28748                               ;
         00000000*  0014A ;TPA$ADDR
                            U.184:  .LONG   <<FPROT_VALUE-U.184>-4>             ;
         00000008  0014E ;TPA$MASK
                            U.185:  .LONG   8                                   ;
             0000*  00152 ;TPA$TARGET
                            U.186:  .WORD   <<SYPRO-U.186>-2>                   ;
             15F6  00154 ;TPA$TYPE
                            U.187:  .WORD   5622                                ;
             0000*  00156 ;TPA$TARGET
                            U.188:  .WORD   <<U.161-U.188>-2>                   ;
                   00158 ;OWPR
                            U.139:  .BLKB   0
             003A  00158 ;TPA$TYPE
                            U.189:  .WORD   58                                  ;
             003D  0015A ;TPA$TYPE
                            U.190:  .WORD   61                                  ;
             15F6  0015C ;TPA$TYPE
                            U.191:  .WORD   5622                                ;
             0000*  0015E ;TPA$TARGET
                            U.192:  .WORD   <<U.161-U.192>-2>                   ;
                   00160 OWPRO:     .BLKB   0
             7052  00160 ;TPA$TYPE
                            U.193:  .WORD   28754                               ;
         00000000*  00162 ;TPA$ADDR
                            U.194:  .LONG   <<FPROT_VALUE-U.194>-4>             ;
         00000010  00166 ;TPA$MASK
                            U.195:  .LONG   16                                  ;
             0000*  0016A ;TPA$TARGET
                            U.196:  .WORD   <<OWPRO-U.196>-2>                   ;
             7057  0016C ;TPA$TYPE
                            U.197:  .WORD   28759                               ;
         00000000*  0016E ;TPA$ADDR
                            U.198:  .LONG   <<FPROT_VALUE-U.198>-4>             ;
         00000020  00172 ;TPA$MASK
```

```
                           U.199:   .LONG    32                              ;
          0000*  00176 ;TPA$TARGET
                           U.200:   .WORD    <<OWPRO-U.200>-2>               ;
          7045   00178 ;TPA$TYPE
                           U.201:   .WORD    28741                           ;
      00000000*  0017A ;TPA$ADDR
                           U.202:   .LONG    <<FPROT_VALUE-U.202>-4>         ;
      00000040   0017E ;TPA$MASK
                           U.203:   .LONG    64                              ;
          0000*  00182 ;TPA$TARGET
                           U.204:   .WORD    <<OWPRO-U.204>-2>               ;
          7050   00184 ;TPA$TYPE
                           U.205:   .WORD    28752                           ;
      00000000*  00186 ;TPA$ADDR
                           U.206:   .LONG    <<FPROT_VALUE-U.206>-4>         ;
      00000040   0018A ;TPA$MASK
                           U.207:   .LONG    64                              ;
          0000*  0018E ;TPA$TARGET
                           U.208:   .WORD    <<OWPRO-U.208>-2>               ;
          7044   00190 ;TPA$TYPE
                           U.209:   .WORD    28740                           ;
      00000000*  00192 ;TPA$ADDR
                           U.210:   .LONG    <<FPROT_VALUE-U.210>-4>         ;
      00000080   00196 ;TPA$MASK
                           U.211:   .LONG    128                             ;
          0000*  0019A ;TPA$TARGET
                           U.212:   .WORD    <<OWPRO-U.212>-2>               ;
          704C   0019C ;TPA$TYPE
                           U.213:   .WORD    28748                           ;
      00000000*  0019E ;TPA$ADDR
                           U.214:   .LONG    <<FPROT_VALUE-U.214>-4>         ;
      00000080   001A2 ;TPA$MASK
                           U.215:   .LONG    128                             ;
          0000*  001A6 ;TPA$TARGET
                           U.216:   .WORD    <<OWPRO-U.216>-2>               ;
          15F6   001A8 ;TPA$TYPE
                           U.217:   .WORD    5622                            ;
          0000*  001AA ;TPA$TARGET
                           U.218:   .WORD    <<U.161-U.218>-2>               ;
                 001AC ;GRPR
                           U.147:   .BLKB    0
          003A   001AC ;TPA$TYPE
                           U.219:   .WORD    58                              ;
          003D   001AE ;TPA$TYPE
                           U.220:   .WORD    61                              ;
          15F6   001B0 ;TPA$TYPE
                           U.221:   .WORD    5622                            ;
          0000*  001B2 ;TPA$TARGET
                           U.222:   .WORD    <<U.161-U.222>-2>               ;
                 001B4 GRPRO:       .BLKB    0
          7052   001B4 ;TPA$TYPE
                           U.223:   .WORD    28754                           ;
      00000000*  001B6 ;TPA$ADDR
                           U.224:   .LONG    <<FPROT_VALUE-U.224>-4>         ;
      00000100   001BA ;TPA$MASK
                           U.225:   .LONG    256                             ;
          0000*  001BE ;TPA$TARGET
```

```
                               U.226:   .WORD   <<GRPRO-U.226>-2>                   ;
            7057  001C0 ;TPA$TYPE
                               U.227:   .WORD   28759                               ;
       00000000*  001C2 ;TPA$ADDR
                               U.228:   .LONG   <<FPROT_VALUE-U.228>-4>             ;
        00000200  001C6 ;TPA$MASK
                               U.229:   .LONG   512                                 ;
           0000*  001CA ;TPA$TARGET
                               U.230:   .WORD   <<GRPRO-U.230>-2>                   ;
            7045  001CC ;TPA$TYPE
                               U.231:   .WORD   28741                               ;
       00000000*  001CE ;TPA$ADDR
                               U.232:   .LONG   <<FPROT_VALUE-U.232>-4>             ;
        00000400  001D2 ;TPA$MASK
                               U.233:   .LONG   1024                                ;
           0000*  001D6 ;TPA$TARGET
                               U.234:   .WORD   <<GRPRO-U.234>-2>                   ;
            7050  001D8 ;TPA$TYPE
                               U.235:   .WORD   28752                               ;
       00000000*  001DA ;TPA$ADDR
                               U.236:   .LONG   <<FPROT_VALUE-U.236>-4>             ;
        00000400  001DE ;TPA$MASK
                               U.237:   .LONG   1024                                ;
           0000*  001E2 ;TPA$TARGET
                               U.238:   .WORD   <<GRPRO-U.238>-2>                   ;
            7044  001E4 ;TPA$TYPE
                               U.239:   .WORD   28740                               ;
       00000000*  001E6 ;TPA$ADDR
                               U.240:   .LONG   <<FPROT_VALUE-U.240>-4>             ;
        00000800  001EA ;TPA$MASK
                               U.241:   .LONG   2048                                ;
           0000*  001EE ;TPA$TARGET
                               U.242:   .WORD   <<GRPRO-U.242>-2>                   ;
            704C  001F0 ;TPA$TYPE
                               U.243:   .WORD   28748                               ;
       00000000*  001F2 ;TPA$ADDR
                               U.244:   .LONG   <<FPROT_VALUE-U.244>-4>             ;
        00000800  001F6 ;TPA$MASK
                               U.245:   .LONG   2048                                ;
           0000*  001FA ;TPA$TARGET
                               U.246:   .WORD   <<GRPRO-U.246>-2>                   ;
            15F6  001FC ;TPA$TYPE
                               U.247:   .WORD   5622                                ;
           0000*  001FE ;TPA$TARGET
                               U.248:   .WORD   <<U.161-U.248>-2>                   ;
                  00200 ;WOPR
                               U.155:   .BLKB   0
            003A  00200 ;TPA$TYPE
                               U.249:   .WORD   58                                  ;
            003D  00202 ;TPA$TYPE
                               U.250:   .WORD   61                                  ;
            15F6  00204 ;TPA$TYPE
                               U.251:   .WORD   5622                                ;
           0000*  00206 ;TPA$TARGET
                               U.252:   .WORD   <<U.161-U.252>-2>                   ;
                  00208 WOPRO:          .BLKB   0
            7052  00208 ;TPA$TYPE
```

```
                    U.253:  .WORD   28754                          ;
00000000*  0020A   ;TPA$ADDR
                    U.254:  .LONG   <<FPROT_VALUE-U.254>-4>        ;
00001000   0020E   ;TPA$MASK
                    U.255:  .LONG   4096                           ;
    0000*  00212   ;TPA$TARGET
                    U.256:  .WORD   <<WOPRO-U.256>-2>              ;
     7057  00214   ;TPA$TYPE
                    U.257:  .WORD   28759                          ;
00000000*  00216   ;TPA$ADDR
                    U.258:  .LONG   <<FPROT_VALUE-U.258>-4>        ;
00002000   0021A   ;TPA$MASK
                    U.259:  .LONG   8192                           ;
    0000*  0021E   ;TPA$TARGET
                    U.260:  .WORD   <<WOPRO-U.260>-2>              ;
     7045  00220   ;TPA$TYPE
                    U.261:  .WORD   28741                          ;
00000000*  00222   ;TPA$ADDR
                    U.262:  .LONG   <<FPROT_VALUE-U.262>-4>        ;
00004000   00226   ;TPA$MASK
                    U.263:  .LONG   16384                          ;
    0000*  0022A   ;TPA$TARGET
                    U.264:  .WORD   <<WOPRO-U.264>-2>              ;
     7050  0022C   ;TPA$TYPE
                    U.265:  .WORD   28752                          ;
00000000*  0022E   ;TPA$ADDR
                    U.266:  .LONG   <<FPROT_VALUE-U.266>-4>        ;
00004000   00232   ;TPA$MASK
                    U.267:  .LONG   16384                          ;
    0000*  00236   ;TPA$TARGET
                    U.268:  .WORD   <<WOPRO-U.268>-2>              ;
     7044  00238   ;TPA$TYPE
                    U.269:  .WORD   28740                          ;
00000000*  0023A   ;TPA$ADDR
                    U.270:  .LONG   <<FPROT_VALUE-U.270>-4>        ;
00008000   0023E   ;TPA$MASK
                    U.271:  .LONG   32768                          ;
    0000*  00242   ;TPA$TARGET
                    U.272:  .WORD   <<WOPRO-U.272>-2>              ;
     704C  00244   ;TPA$TYPE
                    U.273:  .WORD   28748                          ;
00000000*  00246   ;TPA$ADDR
                    U.274:  .LONG   <<FPROT_VALUE-U.274>-4>        ;
00008000   0024A   ;TPA$MASK
                    U.275:  .LONG   32768                          ;
    0000*  0024E   ;TPA$TARGET
                    U.276:  .WORD   <<WOPRO-U.276>-2>              ;
     15F6  00250   ;TPA$TYPE
                    U.277:  .WORD   5622                           ;
    0000*  00252   ;TPA$TARGET
                    U.278:  .WORD   <<U.161-U.278>-2>              ;
           00254   ;ENDPRO
                    U.161:  .BLKB   0
     102C  00254   ;TPA$TYPE
                    U.279:  .WORD   4140                           ;
    0000*  00256   ;TPA$TARGET
                    U.280:  .WORD   <<NEXTPRO-U.280>-2>            ;
```

```
             15F7  00258 ;TPA$TYPE
                         U.281:  .WORD   5623                                    ;
             FFFF  0025A ;TPA$TARGET
                         U.282:  .WORD   -1                                      ;
                   0025C RET_STATE::
                                 .BLKB   0
                   0025C RETSTART:
                                 .BLKB   0
             4DF8  0025C ;TPA$TYPE
                         U.284:  .WORD   19960                                   ;
            0000*  0025E ;TPA$SUBEXP
                         U.286:  .WORD   <<U.285-U.286>-2>                       ;
        00000000*  00260 ;TPA$ADDR
                         U.287:  .LONG   <<RETMIN_VALUE-U.287>-4>                ;
             002C  00264 ;TPA$TYPE
                         U.288:  .WORD   44                                      ;
             15F7  00266 ;TPA$TYPE
                         U.289:  .WORD   5623                                    ;
             FFFF  00268 ;TPA$TARGET
                         U.290:  .WORD   -1                                      ;
             4DF8  0026A ;TPA$TYPE
                         U.291:  .WORD   19960                                   ;
            0000*  0026C ;TPA$SUBEXP
                         U.292:  .WORD   <<U.285-U.292>-2>                       ;
        00000000*  0026E ;TPA$ADDR
                         U.293:  .LONG   <<RETMAX_VALUE-U.293>-4>                ;
             15F7  00272 ;TPA$TYPE
                         U.294:  .WORD   5623                                    ;
             FFFF  00274 ;TPA$TARGET
                         U.295:  .WORD   -1                                      ;
                   00276 ;GET_DELTA
                         U.285:  .BLKB   0
             91ED  00276 ;TPA$TYPE
                         U.296:  .WORD   -28179                                  ;
        00000000V  00278 ;TPA$ACTION
                         U.297:  .LONG   <<TEST_CHAR-U.297>-4>                   ;
            0000*  0027C ;TPA$TARGET
                         U.298:  .WORD   <<U.285-U.298>-2>                       ;
             15F6  0027E ;TPA$TYPE
                         U.299:  .WORD   5622                                    ;
             FFFF  00280 ;TPA$TARGET
                         U.300:  .WORD   -1                                      ;

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

                   00000 DC_KEYS::
                                 .BLKB   0
                   00000 ;TPA$KEY0
                         U.1:    .BLKB   0
            0000*  00000 ;TPA$KEY
                         U.10:   .WORD   <U.9-U.1>                               ;
            0000*  00002 ;TPA$KEY
                         U.16:   .WORD   <U.15-U.1>                              ;
            0000*  00004 ;TPA$KEY
                         U.22:   .WORD   <U.21-U.1>                              ;
            0000*  00006 ;TPA$KEY
                         U.28:   .WORD   <U.27-U.1>                              ;
```

```
                           00008 OWNER_KEYS::
                                            .BLKB   0
                           00008 ;TPA$KEY0
                                 U.38:      .BLKB   0
                     0000* 00008 ;TPA$KEY
                                 U.40:      .WORD   <U.39-U.38>                          ;
                           0000A            .BLKB   2
                           0000C JOURNAL_KEYS::
                                            .BLKB   0
                           0000C ;TPA$KEY0
                                 U.58:      .BLKB   0
                     0000* 0000C ;TPA$KEY
                                 U.60:      .WORD   <U.59-U.58>                          ;
                     0000* 0000E ;TPA$KEY
                                 U.66:      .WORD   <U.65-U.58>                          ;
                     0000* 00010 ;TPA$KEY
                                 U.72:      .WORD   <U.71-U.58>                          ;
                     0000* 00012 ;TPA$KEY
                                 U.78:      .WORD   <U.77-U.58>                          ;
                     0000* 00014 ;TPA$KEY
                                 U.84:      .WORD   <U.83-U.58>                          ;
                     0000* 00016 ;TPA$KEY
                                 U.90:      .WORD   <U.89-U.58>                          ;
                     0000* 00018 ;TPA$KEY
                                 U.96:      .WORD   <U.95-U.58>                          ;
                     0000* 0001A ;TPA$KEY
                                 U.102:     .WORD   <U.101-U.58>                         ;
                     0000* 0001C ;TPA$KEY
                                 U.108:     .WORD   <U.107-U.58>                         ;
                     0000* 0001E ;TPA$KEY
                                 U.114:     .WORD   <U.113-U.58>                         ;
                           00020 PRO_KEYS::
                                            .BLKB   0
                           00020 ;TPA$KEY0
                                 U.124:     .BLKB   0
                     0000* 00020 ;TPA$KEY
                                 U.126:     .WORD   <U.125-U.124>                        ;
                     0000* 00022 ;TPA$KEY
                                 U.134:     .WORD   <U.133-U.124>                        ;
                     0000* 00024 ;TPA$KEY
                                 U.142:     .WORD   <U.141-U.124>                        ;
                     0000* 00026 ;TPA$KEY
                                 U.150:     .WORD   <U.149-U.124>                        ;
                           00028 RET_KEYS::
                                            .BLKB   0
                           00028 ;TPA$KEY0
                                 U.283:     .BLKB   0

                                            .PSECT  $OWN$,NOEXE,2

    00000003   00000008    00000 TPARSE_BLOCK:
                                            .LONG   8, 3                                 ;
                           00008            .BLKB   28
                           00024 PRIVS:     .BLKB   8

                                 QUAL_ACCESS==          1
                                 QUAL_BACKUP==          2
```

```
QUAL_NOBACKUP==         3
QUAL_CONFIRM==          4
QUAL_DATA==             5
QUAL_ENTER==            6
QUAL_EOF==              7
QUAL_ERASE==            8
QUAL_NOERASE==          9
QUAL_EXPI==             10
QUAL_EXTE==             11
QUAL_FPROT==            12
QUAL_GBUF==             13
QUAL_JOURNAL==          14
QUAL_LABEL==            15
QUAL_LOG==              16
QUAL_NODI==             17
QUAL_OWNER==            18
QUAL_PARENT==           19
QUAL_REMOVE==           20
QUAL_RETENT==           21
QUAL_RPROT==            22
QUAL_TRUNC==            23
QUAL_USERNAME==         24
QUAL_VPROT==            25
QUAL_VRSN==             26
QUAL_WINDOWS==          27
DATA_READ==             1
DATA_NOREAD==           2
DATA_WRITE==            3
DATA_NOWRITE==          4
JRNL_AI==               1
JRNL_NOAI==             2
JRNL_AT==               3
JRNL_NOAT==             4
JRNL_BI==               5
JRNL_NOBI==             6
JRNL_RU==               7
JRNL_NORU==             8
JRNL_RUM==              9
JRNL_NORUM==            10
                .EXTRN  CALCULATE_MAX, SYS$FAO
                .EXTRN  LIB$TPARSE, LIB$CVT_TIME
                .EXTRN  LIB$CVT_DTIME, LIB$CVT_DTB
                .EXTRN  RENAME_BUF, FILE_NAME
                .EXTRN  FILE_RLF, SET$L_STATUS
                .EXTRN  SET$A_CLIWORK, SETFILE$FLAGS
                .EXTRN  SETFILE$DFLAGS, SETFILE$JFLAGS
                .EXTRN  ACC_VALUE, EXP_VALUE
                .EXTRN  EXTE_VALUE, FPROT_VALUE
                .EXTRN  GBUF_VALUE, LABEL_VALUE
                .EXTRN  UIC_VALUE, GROUP
                .EXTRN  MEMBER, USER_VALUE
                .EXTRN  RETMIN_VALUE, RETMAX_VALUE
                .EXTRN  VPROT_VALUE, VRSN_VALUE
                .EXTRN  WINDOW_VALUE, SET$_FACILITY
                .EXTRN  SET$_OPERREQ, SET$_WRITEERR
                .EXTRN  SYS$SETPRV
```

```
                                                           .PSECT  $CODE$,NOWRT,2

                                   003C 00000              .ENTRY  ACC_ACT, Save R2,R3,R4,R5                          ; 0396
                55 00000000G   00    9E 00002              MOVAB   ACC_VALUE, R5                                      ;
                54 00000000G   00    9E 00009              MOVAB   LIB$STOP, R4                                       ;
                5E             08    C2 00010              SUBL2   #8, SP                                             ;
                   00000000'   EF    9F 00013              PUSHAB  PRIVS                                              ; 0421
                               01    DD 00019              PUSHL   #1                                                 ;
                7E             01    7D 0001B              MOVQ    #1, -(SP)                                          ;
      00000000G 00             04    FB 0001E              CALLS   #4, SYS$SETPRV                                     ;
                53             50    D0 00025              MOVL    R0, STATUS                                         ;
                05             53    E8 00028              BLBS    STATUS, 1$                                         ;
                               53    DD 0002B              PUSHL   STATUS                                             ; 0422
                64             01    FB 0002D              CALLS   #1, LIB$STOP                                       ;
   09 00000000' EF             02    E0 00030 1$:          BBS     #2, PRIVS+2, 2$                                    ; 0424
                   00000000G   8F    DD 00038              PUSHL   #SET$_OPERREQ                                      ;
                64             01    FB 0003E              CALLS   #1, LIB$STOP                                       ;
                65             03    D0 00041 2$:          MOVL    #3, ACC_VALUE                                      ; 0430
                52          04 AC    D0 00044              MOVL    OPTION_BLOCK, R2                                   ; 0435
                            04 A2    B5 00048              TSTW    4(R2)                                              ;
                               3C    13 0004B              BEQL    6$                                                 ;
                               55    DD 0004D              PUSHL   R5                                                 ; 0441
                            08 A2    DD 0004F              PUSHL   8(R2)                                              ; 0442
                7E          04 A2    3C 00052              MOVZWL  4(R2), -(SP)                                       ; 0441
      00000000G 00             03    FB 00056              CALLS   #3, LIB$CVT_DTB                                    ;
                53             50    D0 0005D              MOVL    R0, STATUS                                         ;
                04             53    E8 00060              BLBS    STATUS, 3$                                         ;
                               53    DD 00063              PUSHL   STATUS                                             ; 0447
                               14    11 00065              BRB     5$                                                 ; 0446
                50             65    D0 00067 3$:          MOVL    ACC_VALUE, R0                                      ; 0450
                               09    19 0006A              BLSS    4$                                                 ;
      000000FF  8F             50    D1 0006C              CMPL    R0, #255                                           ; 0452
                               14    15 00073              BLEQ    6$                                                 ;
                   00000000*   8F    DD 00075 4$:          PUSHL   #<<<SET$_FACILITY@16>+4584>+2>                     ; 0456
                            04 A2    9F 0007B 5$:          PUSHAB  4(R2)                                              ; 0455
                               01    DD 0007E              PUSHL   #1                                                 ;
                   00000000*   8F    DD 00080              PUSHL   #<<<SET$_FACILITY@16>+4344>+2>                     ;
                64             04    FB 00086              CALLS   #4, LIB$STOP                                       ;
                50             01    D0 00089 6$:          MOVL    #1, R0                                             ; 0458
                                     04 0008C              RET                                                        ; 0459

; Routine Size:  141 bytes,    Routine Base:  $CODE$ + 0000
;  472          0460  1 GLOBAL ROUTINE back_act =
;  473          0461  1 !++
;  474          0462  1 !
;  475          0463  1 ! This is the action routine for the /BACKUP qualifier.  It simply
;  476          0464  1 ! sets the correct bit in the flags word.
;  477          0465  1 !
;  478          0466  1 !--
;  479          0467  2 BEGIN
;  480          0468  2 setfile$flags[qual_backup] = true;
;  481          0469  2 RETURN true;
;  482          0470  1 END;



                                          0000 00000          .ENTRY  BACK_ACT, Save nothing                          ; 0460
                   00000000G  00            04  88 00002          BISB2   #4, SETFILE$FLAGS                               ; 0468
                              50            01  D0 00009          MOVL    #1, R0                                          ; 0469
                                              04 0000C          RET                                                     ; 0470

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 008D
;  484       0471  1 GLOBAL ROUTINE noback_act =
;  485       0472  1 !++
;  486       0473  1 !
;  487       0474  1 !  This is the action routine for the /NOBACKUP qualifier.  It simply
;  488       0475  1 !  sets the correct bit in the flags word.
;  489       0476  1 !
;  490       0477  1 !--
;  491       0478  2 BEGIN
;  492       0479  2 setfile$flags[qual_nobackup] = true;
;  493       0480  2 RETURN true;
;  494       0481  1 END;


                                          0000 00000       .ENTRY  NOBACK_ACT, Save nothing                    ; 0471
                  00000000G  00       08   88 00002       BISB2   #8, SETFILE$FLAGS                           ; 0479
                             50       01   D0 00009       MOVL    #1, R0                                      ; 0480
                                           04 0000C       RET                                                 ; 0481

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 009A
;   496          0482  1 GLOBAL ROUTINE data_act (option_block,callback) =
;   497          0483  1 !++
;   498          0484  1 !
;   499          0485  1 ! This is the action routine for the /DATA_CHECK qualifier.  It checks to see
;   500          0486  1 ! if any options were set.  If not, it defaults to DATA_CHECK=WRITE.
;   501          0487  1 !
;   502          0488  1 !--
;   503          0489  2 BEGIN
;   504          0490  2
;   505          0491  2 LOCAL
;   506          0492  2     status;
;   507          0493  2
;   508          0494  2 MAP
;   509          0495  2     option_block : REF BBLOCK;
;   510          0496  2
;   511          0497  2 IF .option_block[cli$w_qdvalsiz] EQL 0
;   512          0498  2 THEN setfile$dflags[data_write] = true
;   513          0499  2
;   514          0500  2 ELSE
;   515          0501  3     BEGIN
;   516          0502  3     tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
;   517          0503  3     tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
;   518          0504  4     IF NOT (status = lib$tparse(tparse_block,
;   519          0505  4             dc_state,dc_keys))
;   520          0506  3     THEN
;   521          0507  4         BEGIN
;   522          0508  4         SIGNAL( set$_facility^16 + shr$_syntax + sts$k_error,
;   523          0509  4                 1,
;   524          0510  4                 option_block[cli$q_qdvaldesc],
;   525          0511  4                 .status);
;   526          0512  4         RETURN .status;
;   527          0513  3         END;
;   528          0514  2     END;
;   529          0515  2 RETURN true;
;   530          0516  1 END;



                                       001C 00000          .ENTRY  DATA_ACT, Save R2,R3,R4                          ; 0482
                 54 00000000'   EF   9E 00002          MOVAB   TPARSE_BLOCK+8, R4
                 52         04  AC   D0 00009          MOVL    OPTION_BLOCK, R2                                 ; 0497
                            04  A2   B5 0000D          TSTW    4(R2)                                            ;
                                09   12 00010          BNEQ    1$                                               ;
           00000000G  00        08   88 00012          BISB2   #8, SETFILE$DFLAGS                               ; 0498
                                3D   11 00019          BRB     2$                                               ;
                 64         04  A2   3C 0001B 1$:      MOVZWL  4(R2), TPARSE_BLOCK+8                            ; 0502
           04    A4         08  A2   D0 0001F          MOVL    8(R2), TPARSE_BLOCK+12                           ; 0503
                    00000000'   EF   9F 00024          PUSHAB  DC_KEYS                                          ; 0504
                    00000000'   EF   9F 0002A          PUSHAB  DC_STATE                                         ;
                            F8  A4   9F 00030          PUSHAB  TPARSE_BLOCK                                     ;
           00000000G  00        03   FB 00033          CALLS   #3, LIB$TPARSE                                   ;
                 53             50   D0 0003A          MOVL    R0, STATUS                                       ;
                 18             53   E8 0003D          BLBS    STATUS, 2$                                       ;
                                53   DD 00040          PUSHL   STATUS                                           ; 0511
                            04  A2   9F 00042          PUSHAB  4(R2)                                            ; 0510
```

```
                                           01   DD 00045          PUSHL    #1                                                  ;
                                00000000*  8F   DD 00047          PUSHL    #<<<SET$_FACILITY@16>+4344>+2>                      ;
                       00000000G  00       04   FB 0004D          CALLS    #4, LIB$SIGNAL                                      ;
                                  50       53   D0 00054          MOVL     STATUS, R0                                          ; 0512
                                                04 00057          RET                                                          ;
                                  50       01   D0 00058  2$:     MOVL     #1, R0                                              ; 0515
                                                04 0005B          RET                                                          ; 0516

; Routine Size:  92 bytes,    Routine Base:  $CODE$ + 00A7
```

```
0517   1 GLOBAL ROUTINE enter_act (option_block,callback) =
0518   1 !++
0519   1 !
0520   1 ! This is the action routine for the /ENTER qualifier.
0521   1 ! The new synonym is collected.
0522   1 !
0523   1 !--
0524   2 BEGIN
0525   2
0526   2 MAP
0527   2     option_block : REF BBLOCK;
0528   2
0529   2
0530   2 !
0531   2 ! Get the expanded file string
0532   2 !
0533   2
0534   2 CH$MOVE(.option_block[cli$w_qdvalsiz],                        ! Move file string
0535   2         .option_block[cli$a_qdvaladr],
0536   2         rename_buf);
0537   2
0538   2 file_name[0] = .option_block[cli$w_qdvalsiz];                 ! Store length
0539   2 file_name[1] = .option_block[cli$a_qdvaladr];                 ! and address
0540   2
0541   2 RETURN true;
0542   1 END;



                                          007C 00000       .ENTRY  ENTER_ACT, Save R2,R3,R4,R5,R6        ; 0517
                          56        04 AC   D0 00002       MOVL    OPTION_BLOCK, R6                      ; 0534
      00000000G  00   08  B6        04 A6   28 00006       MOVC3   4(R6), @8(R6), RENAME_BUF             ;
            00000000G  00           04 A6   3C 00010       MOVZWL  4(R6), FILE_NAME                      ; 0538
            00000000G  00           08 A6   D0 00018       MOVL    8(R6), FILE_NAME+4                    ; 0539
                          50           01   D0 00020       MOVL    #1, R0                                ; 0541
                                           04 00023       RET                                           ; 0542

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0103
;  559          0543   1 GLOBAL ROUTINE erase_act =
;  560          0544   1 !++
;  561          0545   1 !
;  562          0546   1 ! This is the action routine for the /ERASE qualifier.  It simply
;  563          0547   1 ! sets the correct bit in the flags word.
;  564          0548   1 !
;  565          0549   1 !--
;  566          0550   2 BEGIN
;  567          0551   2 setfile$flags[qual_erase] = true;
;  568          0552   2 RETURN true;
;  569          0553   1 END;



                                           0000 00000          .ENTRY  ERASE_ACT, Save nothing                         ; 0543
                      00000000G  00           01   88 00002          BISB2   #1, SETFILE$FLAGS+1                             ; 0551
                                 50           01   D0 00009          MOVL    #1, R0                                          ; 0552
                                                   04 0000C          RET                                                     ; 0553

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 0127
;  571          0554  1 GLOBAL ROUTINE noerase_act =
;  572          0555  1 !++
;  573          0556  1 !
;  574          0557  1 ! This is the action routine for the /NOERASE qualifier.  It simply
;  575          0558  1 ! sets the correct bit in the flags word.
;  576          0559  1 !
;  577          0560  1 !--
;  578          0561  2 BEGIN
;  579          0562  2 setfile$flags[qual_noerase] = true;
;  580          0563  2 RETURN true;
;  581          0564  1 END;



                                                    0000 00000          .ENTRY  NOERASE_ACT, Save nothing                      ; 0554
                    00000000G  00                02   88 00002          BISB2   #2, SETFILE$FLAGS+1                            ; 0562
                               50                01   D0 00009          MOVL    #1, R0                                         ; 0563
                                                      04 0000C          RET                                                    ; 0564

; Routine Size:  13 bytes,    Routine Base:  $CODE$ + 0134
;  583         0565  1 GLOBAL ROUTINE exp_act (option_block,callback) =
;  584         0566  1 !++
;  585         0567  1 !
;  586         0568  1 ! This is the action routine for the /EXPIRATION qualifier.
;  587         0569  1 ! If no value is given, exit with a syntax error.
;  588         0570  1 !
;  589         0571  1 !--
;  590         0572  2 BEGIN
;  591         0573  2
;  592         0574  2 LOCAL
;  593         0575  2     status,
;  594         0576  2     desc : BBLOCK[dsc$c_s_bln];
;  595         0577  2
;  596         0578  2 MAP
;  597         0579  2     option_block : REF BBLOCK;            ! Define the CLI options block
;  598         0580  2
;  599         0581  2 !
;  600         0582  2 ! Get the date, signalling a syntax error if no good.
;  601         0583  2 !
;  602         0584  2
;  603         0585  2 desc[dsc$w_length] = .option_block[cli$w_qdvalsiz];
;  604         0586  2 desc[dsc$a_pointer] = .option_block[cli$a_qdvaladr];
;  605         0587  3 IF NOT (status = LIB$CVT_TIME(desc,exp_value))
;  606         0588  2 THEN
;  607         0589  3     BEGIN
;  608         0590  3     SIGNAL_STOP(  set$_facility^16 + shr$_syntax + sts$k_error,
;  609         0591  3                   1,
;  610         0592  3                   option_block[cli$q_qdvaldesc],
;  611         0593  3                   .status);
;  612         0594  3     RETURN .status;
;  613         0595  3     END
;  614         0596  2 ELSE RETURN true;
;  615         0597  1 END;



                                    000C 00000          .ENTRY  EXP_ACT, Save R2,R3                          ; 0565
                   5E            08   C2 00002          SUBL2   #8, SP                                       ;
                   52       04   AC   D0 00005          MOVL    OPTION_BLOCK, R2                             ; 0585
                   6E       04   A2   B0 00009          MOVW    4(R2), DESC                                  ;
             04    AE       08   A2   D0 0000D          MOVL    8(R2), DESC+4                                ; 0586
                      00000000G  00   9F 00012          PUSHAB  EXP_VALUE                                    ; 0587
                            04   AE   9F 00018          PUSHAB  DESC                                         ;
         00000000G  00           02   FB 0001B          CALLS   #2, LIB$CVT_TIME                             ;
                   53            50   D0 00022          MOVL    R0, STATUS                                   ;
                   18            53   E8 00025          BLBS    STATUS, 1$                                   ;
                                 53   DD 00028          PUSHL   STATUS                                       ; 0593
                            04   A2   9F 0002A          PUSHAB  4(R2)                                        ; 0592
                                 01   DD 0002D          PUSHL   #1                                           ;
                      00000000*  8F   DD 0002F          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>               ;
         00000000G  00           04   FB 00035          CALLS   #4, LIB$STOP                                 ;
                   50            53   D0 0003C          MOVL    STATUS, R0                                   ; 0596
                                      04 0003F          RET                                                  ;
                   50            01   D0 00040 1$:      MOVL    #1, R0                                       ;
                                      04 00043          RET                                                  ; 0597
```

```
; Routine Size:  68 bytes,    Routine Base:  $CODE$ + 0141
;  617          0598  1 GLOBAL ROUTINE noexp_act =
;  618          0599  1 !++
;  619          0600  1 !
;  620          0601  1 ! This is the action routine for the /NOEXPIRATION_DATE qualifier.
;  621          0602  1 ! It supplies an expiration date of zero.
;  622          0603  1 !
;  623          0604  1 !--
;  624          0605  2 BEGIN
;  625          0606  2
;  626          0607  2 CH$FILL(0,8,exp_value);                   ! Zero out the expiration date
;  627          0608  2 setfile$flags[qual_expi] = true;          ! Set the expiration flag on
;  628          0609  2
;  629          0610  2 RETURN true;
;  630          0611  1 END;



                                          003C 00000          .ENTRY  NOEXP_ACT, Save R2,R3,R4,R5                 ; 0598
     08         00           6E        00   2C 00002          MOVC5   #0, (SP), #0, #8, EXP_VALUE                 ; 0607
                                  00000000G  00    00007                                                           ;
                   00000000G  00            04   88 0000C          BISB2   #4, SETFILE$FLAGS+1                         ; 0608
                              50            01   D0 00013          MOVL    #1, R0                                      ; 0610
                                                 04 00016          RET                                                 ; 0611

; Routine Size:  23 bytes,    Routine Base:  $CODE$ + 0185
;  632          0612  1 GLOBAL ROUTINE ext_act (option_block,callback) =
;  633          0613  1 !++
;  634          0614  1 !
;  635          0615  1 ! This is the action routine for the /EXTENSION qualifier.  If no value is
;  636          0616  1 ! specified, the default value of 0 is used.
;  637          0617  1 !
;  638          0618  1 !--
;  639          0619  2 BEGIN
;  640          0620  2
;  641          0621  2 LOCAL
;  642          0622  2     status,
;  643          0623  2     desc : BBLOCK[dsc$c_s_bln];
;  644          0624  2
;  645          0625  2 MAP
;  646          0626  2     option_block : REF BBLOCK;            ! Define the CLI options block
;  647          0627  2
;  648          0628  2 exte_value = 0;                           ! Set up default
;  649          0629  2
;  650          0630  2 !
;  651          0631  2 ! See if a value was specified.  If not, then use the default.
;  652          0632  2 !
;  653          0633  2 IF .option_block[cli$w_qdvalsiz] EQL 0
;  654          0634  2 THEN RETURN true;
;  655          0635  2
;  656          0636  2 !
;  657          0637  2 ! If the value is there, convert it and return
;  658          0638  2 !
;  659          0639  3 IF NOT (status = LIB$CVT_DTB(.option_block[cli$w_qdvalsiz],
;  660          0640  3                              .option_block[cli$a_qdvaladr],
;  661          0641  3                              exte_value))
;  662          0642  2 THEN
;  663          0643  3     BEGIN
;  664          0644  3     SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  665          0645  3                 1,
;  666          0646  3                 option_block[cli$q_qdvaldesc],
;  667          0647  3                 .status);
;  668          0648  3     END
;  669          0649  2 ELSE
;  670          0650  3     BEGIN
;  671          0651  4     IF NOT (.exte_value GEQ 0
;  672          0652  4                 AND
;  673          0653  4             .exte_value LEQ 65535)
;  674          0654  3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  675          0655  3                      1,
;  676          0656  3                      option_block[cli$q_qdvaldesc],
;  677          0657  3                      set$_facility^16 + shr$_valerr + sts$k_error);
;  678          0658  2     END;
;  679          0659  2
;  680          0660  2 RETURN true;
;  681          0661  1 END;




                                          000C 00000           .ENTRY  EXT_ACT, Save R2,R3                              ; 0612
                      53 00000000G   00   9E 00002             MOVAB   EXTE_VALUE, R3                                   ;
```

```
                  5E              08  C2 00009         SUBL2   #8, SP                                 ;
                                  63  D4 0000C         CLRL    EXTE_VALUE                             ; 0628
                  52        04    AC  D0 0000E         MOVL    OPTION_BLOCK, R2                       ; 0633
                            04    A2  B5 00012         TSTW    4(R2)                                  ;
                                  3D  13 00015         BEQL    4$                                     ;
                                  53  DD 00017         PUSHL   R3                                     ; 0639
                            08    A2  DD 00019         PUSHL   8(R2)                                  ; 0640
                  7E        04    A2  3C 0001C         MOVZWL  4(R2), -(SP)                           ; 0639
         00000000G  00            03  FB 00020         CALLS   #3, LIB$CVT_DTB                        ;
                  04              50  E8 00027         BLBS    STATUS, 1$                             ;
                                  50  DD 0002A         PUSHL   STATUS                                 ; 0647
                                  14  11 0002C         BRB     3$                                     ; 0646
                  50              63  D0 0002E 1$:     MOVL    EXTE_VALUE, R0                         ; 0651
                                  09  19 00031         BLSS    2$                                     ;
         0000FFFF   8F            50  D1 00033         CMPL    R0, #65535                             ; 0653
                                  18  15 0003A         BLEQ    4$                                     ;
                    00000000*     8F  DD 0003C 2$:     PUSHL   #<<<SET$_FACILITY@16>+4584>+2>         ; 0657
                            04    A2  9F 00042 3$:     PUSHAB  4(R2)                                  ; 0656
                                  01  DD 00045         PUSHL   #1                                     ;
                    00000000*     8F  DD 00047         PUSHL   #<<<SET$_FACILITY@16>+4344>+2>         ;
         00000000G  00            04  FB 0004D         CALLS   #4, LIB$STOP                           ;
                  50              01  D0 00054 4$:     MOVL    #1, R0                                 ; 0660
                                      04 00057         RET                                            ; 0661

; Routine Size:  88 bytes,    Routine Base:  $CODE$ + 019C
;  683          0662  1 GLOBAL ROUTINE fprot_act (option_block,callback) =
;  684          0663  1 !++
;  685          0664  1 !
;  686          0665  1 ! This is the action routine for the /FILE_PROTECTION qualifier of
;  687          0666  1 ! SET VOLUME.  The protection is parsed and stored away.  If the
;  688          0667  1 ! protection is not valid, a fatal error message is issued.
;  689          0668  1 !
;  690          0669  1 !--
;  691          0670  2 BEGIN
;  692          0671  2
;  693          0672  2 LOCAL status;                             ! Status return
;  694          0673  2
;  695          0674  2 MAP option_block : REF BBLOCK;            ! Define the option block
;  696          0675  2
;  697          0676  2 !
;  698          0677  2 ! Stuff the TPARSE block with the string
;  699          0678  2 !
;  700          0679  2 tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
;  701          0680  2 tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
;  702          0681  2
;  703          0682  2 fprot_value = 0;                          ! Initialize file protection value
;  704          0683  2
;  705          0684  2 !
;  706          0685  2 ! Now to parse the protection given.  When finished, FPROT_VALUE will
;  707          0686  2 ! have the following values:
;  708          0687  2 !
;  709          0688  2 ! FPROT_VALUE[low_word] = protection value
;  710          0689  2 ! FPROT_VALUE[high_word] = group mask i.e. SYSTEM, OWNER, GROUP, WORLD
;  711          0690  2 !
;  712          0691  3 IF NOT (status = LIB$TPARSE(tparse_block,
;  713          0692  3                         pro_state,
;  714          0693  3                         pro_keys))
;  715          0694  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  716          0695  2                 1,
;  717          0696  2                 option_block[cli$q_qdvaldesc],
;  718          0697  2                 .status);
;  719          0698  2 RETURN true;
;  720          0699  1 END;



                        000C 00000          .ENTRY  FPROT_ACT, Save R2,R3                          ; 0662
     53 00000000'   EF   9E 00002          MOVAB   TPARSE_BLOCK+8, R3                             ;
     52       04    AC   D0 00009          MOVL    OPTION_BLOCK, R2                               ; 0679
     63       04    A2   3C 0000D          MOVZWL  4(R2), TPARSE_BLOCK+8                          ;
04   A3       08    A2   D0 00011          MOVL    8(R2), TPARSE_BLOCK+12                         ; 0680
        00000000G   00   D4 00016          CLRL    FPROT_VALUE                                    ; 0682
        00000000'   EF   9F 0001C          PUSHAB  PRO_KEYS                                       ; 0691
        00000000'   EF   9F 00022          PUSHAB  PRO_STATE                                      ;
              F8    A3   9F 00028          PUSHAB  TPARSE_BLOCK                                   ;
00000000G  00       03   FB 0002B          CALLS   #3, LIB$TPARSE                                 ;
           14       50   E8 00032          BLBS    STATUS, 1$                                     ;
                    50   DD 00035          PUSHL   STATUS                                         ; 0697
              04    A2   9F 00037          PUSHAB  4(R2)                                          ; 0696
                    01   DD 0003A          PUSHL   #1                                             ;
```

```
                              00000000*  8F  DD 0003C         PUSHL   #<<<SET$_FACILITY@16>+4344>+2>                 ;
                  00000000G  00          04  FB 00042         CALLS   #4, LIB$STOP                                   ;
                             50          01  D0 00049 1$:     MOVL    #1, R0                                         ; 0698
                                             04 0004C         RET                                                    ; 0699

; Routine Size:  77 bytes,    Routine Base:  $CODE$ + 01F4
;  722          0700  1 GLOBAL ROUTINE gbuf_act (option_block,callback) =
;  723          0701  1 !++
;  724          0702  1 !
;  725          0703  1 ! This is the action routine for the GLOBAL_BUFFER qualifier.  The number of
;  726          0704  1 ! global buffers desired is collected.
;  727          0705  1 !
;  728          0706  1 !--
;  729          0707  2 BEGIN
;  730          0708  2
;  731          0709  2 LOCAL
;  732          0710  2     status,
;  733          0711  2     desc : BBLOCK[dsc$c_s_bln];
;  734          0712  2
;  735          0713  2 MAP
;  736          0714  2     option_block : REF BBLOCK;          ! Define the CLI options block
;  737          0715  2
;  738          0716  2 !
;  739          0717  2 ! Convert the value given (in ASCII) to a numeric value.
;  740          0718  2 !
;  741          0719  3 IF NOT (status = LIB$CVT_DTB(.option_block[cli$w_qdvalsiz],
;  742          0720  3                             .option_block[cli$a_qdvaladr],
;  743          0721  3                             gbuf_value))
;  744          0722  2 THEN
;  745          0723  3     BEGIN
;  746          0724  3     SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  747          0725  3                 1,
;  748          0726  3                 option_block[cli$q_qdvaldesc],
;  749          0727  3                 .status);
;  750          0728  3     END
;  751          0729  2 ELSE
;  752          0730  3     BEGIN
;  753          0731  3 !
;  754          0732  3 ! If the value is not a word or less in length, signal an error.
;  755          0733  3 !
;  756          0734  4     IF NOT (.gbuf_value GEQ 0 AND .gbuf_value LEQ 65535)
;  757          0735  3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  758          0736  3                      1,
;  759          0737  3                      option_block[cli$q_qdvaldesc],
;  760          0738  3                      set$_facility^16 + shr$_valerr + sts$k_error);
;  761          0739  2     END;
;  762          0740  2
;  763          0741  2 RETURN true;
;  764          0742  1 END;



                                   000C 00000          .ENTRY  GBUF_ACT, Save R2,R3                   ; 0700
              53 00000000G    00   9E 00002          MOVAB   GBUF_VALUE, R3                         ;
              5E              08   C2 00009          SUBL2   #8, SP                                 ;
                              53   DD 0000C          PUSHL   R3                                     ; 0719
              52        04    AC   D0 0000E          MOVL    OPTION_BLOCK, R2                       ; 0720
                        08    A2   DD 00012          PUSHL   8(R2)                                  ;
              7E        04    A2   3C 00015          MOVZWL  4(R2), -(SP)                           ; 0719
      00000000G 00            03   FB 00019          CALLS   #3, LIB$CVT_DTB                        ;
                04            50   E8 00020          BLBS    STATUS, 1$                             ;
```

```
                                50  DD 00023            PUSHL    STATUS                                  ; 0727
                                14  11 00025            BRB      3$                                      ; 0726
                   50           63  D0 00027 1$:        MOVL     GBUF_VALUE, R0                          ; 0734
                                09  19 0002A            BLSS     2$                                      ;
          0000FFFF 8F           50  D1 0002C            CMPL     R0, #65535                              ;
                                18  15 00033            BLEQ     4$                                      ;
                     00000000*  8F  DD 00035 2$:        PUSHL    #<<<SET$_FACILITY@16>+4584>+2>          ; 0738
                            04  A2  9F 0003B 3$:        PUSHAB   4(R2)                                   ; 0737
                                01  DD 0003E            PUSHL    #1                                      ;
                     00000000*  8F  DD 00040            PUSHL    #<<<SET$_FACILITY@16>+4344>+2>          ;
          00000000G 00          04  FB 00046            CALLS    #4, LIB$STOP                            ;
                    50          01  D0 0004D 4$:        MOVL     #1, R0                                  ; 0741
                                    04 00050            RET                                              ; 0742

; Routine Size:  81 bytes,    Routine Base:  $CODE$ + 0241
;  766        0743  1 GLOBAL ROUTINE journal_act (option_block,callback) =
;  767        0744  1 !++
;  768        0745  1 !
;  769        0746  1 ! This is the action routine for the /JOURNAL qualifier.  Based on the
;  770        0747  1 ! journal types set, specific journaling bits are set.
;  771        0748  1 !
;  772        0749  1 !--
;  773        0750  2 BEGIN
;  774        0751  2
;  775        0752  2 LOCAL
;  776        0753  2     status;
;  777        0754  2
;  778        0755  2 MAP
;  779        0756  2     option_block : REF BBLOCK;
;  780        0757  2
;  781        0758  2 !
;  782        0759  2 ! Use TPARSE to parse the journal list.
;  783        0760  2 !
;  784        0761  2
;  785        0762  2 tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
;  786        0763  2 tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
;  787        0764  2
;  788        0765  3 IF NOT (status = LIB$TPARSE(tparse_block, journal_state, journal_keys))
;  789        0766  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  790        0767  2                 1,
;  791        0768  2                 option_block[cli$q_qdvaldesc],
;  792        0769  2                 .status);
;  793        0770  2 !
;  794        0771  2 ! If both RU and RUM were specified, then signal a syntax error.
;  795        0772  2 !
;  796        0773  2
;  797        0774  3 IF (.setfile$jflags[jrnl_ru] AND .setfile$jflags[jrnl_rum])
;  798        0775  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  799        0776  2                 1,
;  800        0777  2                 option_block[cli$q_qdvaldesc],
;  801        0778  2                 set$_facility^16 + shr$_confqual + sts$k_error);
;  802        0779  2
;  803        0780  2 RETURN true;
;  804        0781  1 END;



                                 001C 00000        .ENTRY   JOURNAL_ACT, Save R2,R3,R4                        ; 0743
            54 00000000G    00     9E 00002        MOVAB    LIB$STOP, R4                                      ;
            53 00000000'    EF     9E 00009        MOVAB    TPARSE_BLOCK+8, R3                                ;
            52          04  AC     D0 00010        MOVL     OPTION_BLOCK, R2                                  ; 0762
            63          04  A2     3C 00014        MOVZWL   4(R2), TPARSE_BLOCK+8                             ;
      04    A3          08  A2     D0 00018        MOVL     8(R2), TPARSE_BLOCK+12                            ; 0763
               00000000'    EF     9F 0001D        PUSHAB   JOURNAL_KEYS                                      ; 0765
               00000000'    EF     9F 00023        PUSHAB   JOURNAL_STATE                                     ;
                        F8  A3     9F 00029        PUSHAB   TPARSE_BLOCK                                      ;
 00000000G  00                     03 FB 0002C     CALLS    #3, LIB$TPARSE                                    ;
            10                     50 E8 00033     BLBS     STATUS, 1$                                        ;
                                   50 DD 00036     PUSHL    STATUS                                            ; 0769
                        04  A2     9F 00038        PUSHAB   4(R2)                                             ; 0768
```

```
                                          01  DD 0003B          PUSHL   #1                                          ;
                             00000000*    8F  DD 0003D          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>              ;
                        64                04  FB 00043          CALLS   #4, LIB$STOP                                ;
                             00000000G    00  95 00046 1$:      TSTB    SETFILE$JFLAGS                              ; 0774
                                          1C  18 0004C          BGEQ    2$                                          ;
               14 00000000G  00           01  E1 0004E          BBC     #1, SETFILE$JFLAGS+1, 2$                    ;
                             00000000*    8F  DD 00056          PUSHL   #<<<SET$_FACILITY@16>+4832>+2>              ; 0778
                                    04    A2  9F 0005C          PUSHAB  4(R2)                                       ; 0777
                                          01  DD 0005F          PUSHL   #1                                          ;
                             00000000*    8F  DD 00061          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>              ;
                        64                04  FB 00067          CALLS   #4, LIB$STOP                                ;
                        50                01  D0 0006A 2$:      MOVL    #1, R0                                      ; 0780
                                              04 0006D          RET                                                 ; 0781

; Routine Size:  110 bytes,    Routine Base:  $CODE$ + 0292
;  806          0782  1 GLOBAL ROUTINE label_act (option_block, callback) =
;  807          0783  1 !++
;  808          0784  1 !
;  809          0785  1 ! This is the action routine for the LABEL qualifier of SET VOLUME.  It
;  810          0786  1 ! retrieves the value of the string, checks that it is no longer than
;  811          0787  1 ! twelve characters, and stores length and location in LABEL_VALUE.
;  812          0788  1 !
;  813          0789  1 !--
;  814          0790  2 BEGIN
;  815          0791  2
;  816          0792  2 LOCAL status;                           ! Status return
;  817          0793  2
;  818          0794  2 MAP option_block : REF BBLOCK;          ! Define the cli block
;  819          0795  2
;  820          0796  2 !
;  821          0797  2 ! Check that the string is no longer than twelve characters.
;  822          0798  2 !
;  823          0799  2 IF .option_block[cli$w_qdvalsiz] GTR 12
;  824          0800  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
;  825          0801  2                 1,
;  826          0802  2                 option_block[cli$q_qdvaldesc],
;  827          0803  2                 set$_facility^16 + shr$_valerr + sts$k_error);
;  828          0804  2 !
;  829          0805  2 ! Store the location and length in LABEL_VALUE
;  830          0806  2 !
;  831          0807  2 label_value[0] = .option_block[cli$w_qdvalsiz];
;  832          0808  2 label_value[1] = .option_block[cli$a_qdvaladr];
;  833          0809  2
;  834          0810  2 RETURN true;
;  835          0811  1 END;



                                   0004 00000          .ENTRY  LABEL_ACT, Save R2                       ; 0782
                 52          04  AC  D0 00002          MOVL    OPTION_BLOCK, R2                         ; 0799
                 0C          04  A2  B1 00006          CMPW    4(R2), #12                               ;
                                 18  1B 0000A          BLEQU   1$                                       ;
                    00000000*    8F  DD 0000C          PUSHL   #<<<SET$_FACILITY@16>+4584>+2>           ; 0803
                             04  A2  9F 00012          PUSHAB  4(R2)                                    ; 0802
                                 01  DD 00015          PUSHL   #1                                       ;
                    00000000*    8F  DD 00017          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>           ;
       00000000G 00              04  FB 0001D          CALLS   #4, LIB$STOP                             ;
       00000000G 00          04  A2  3C 00024 1$:      MOVZWL  4(R2), LABEL_VALUE                       ; 0807
       00000000G 00          08  A2  D0 0002C          MOVL    8(R2), LABEL_VALUE+4                     ; 0808
                 50              01  D0 00034          MOVL    #1, R0                                   ; 0810
                                     04 00037          RET                                              ; 0811

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 0300
```

```
  1 GLOBAL ROUTINE owner_act (option_block,callback) =
  1 !++
  1 !
  1 ! This is the action routine for the OWNER_UIC qualifier.  The input is
  1 ! parsed to obtain the group and member numbers of the UIC.
  1 !
  1 !--
  2 BEGIN
  2
  2 LOCAL
  2     status;                                 ! Status
  2
  2 MAP
  2     option_block : REF BBLOCK;
  2
  2 uic_value = 0;                              ! Set the UIC value to zero initially
  2
  2 !
  2 ! Check to see if UIC specified.  If not, use current process UIC.
  2 !
  2 IF .option_block[cli$w_qdvalsiz] EQL 0
P 2 THEN $GETJPI(ITMLST = UPLIT(WORD(4,jpi$_uic),
P 2                                 uic_value,
P 2                                 0,
  3                                 0))
  2 ELSE
  3     BEGIN
  3     tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
  3     tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
  4     IF NOT (status = lib$tparse(tparse_block,
  4                                 owner_state,
  4                                 owner_keys))
  3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
  3                     1,
  3                     option_block[cli$q_qdvaldesc]);
  3     IF NOT .setfile$flags[qual_parent]
  3     THEN
  4         BEGIN
  6         IF NOT ((.group LEQ %O'377' AND .group GEQ 0)
  5                 AND
  5             (.member LEQ %O'377' AND .member GEQ 0))
  4         THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
  4                         1,
  4                         option_block[cli$q_qdvaldesc],
  4                         set$_facility^16 + shr$_valerr + sts$k_error)
  4         ELSE uic_value = .group^16 + .member;
  3         END;
  2     END;
  2
  2 RETURN true;
  1 END;


                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                          0304  0004  00000 P.AAA:  .WORD   4, 772                          ;
```

```
                                            00000000G 00004       .ADDRESS UIC_VALUE                              ;
                              00000000   00000000  00008       .LONG    0, 0                                   ;

                                                               .EXTRN   SYS$GETJPI

                                                               .PSECT   $CODE$,NOWRT,2

                                                  007C 00000       .ENTRY   OWNER_ACT, Save R2,R3,R4,R5,R6          ; 0812
                           56 00000000G  00   9E 00002       MOVAB    UIC_VALUE, R6                          ;
                           55 00000000G  00   9E 00009       MOVAB    MEMBER, R5                             ;
                           54 00000000G  00   9E 00010       MOVAB    LIB$STOP, R4                           ;
                           53 00000000'  EF   9E 00017       MOVAB    TPARSE_BLOCK+8, R3                     ;
                                         66   D4 0001E       CLRL     UIC_VALUE                              ; 0827
                           52        04  AC   D0 00020       MOVL     OPTION_BLOCK, R2                       ; 0832
                                     04  A2   B5 00024       TSTW     4(R2)                                  ;
                                         17   12 00027       BNEQ     1$                                     ;
                                         7E   7C 00029       CLRQ     -(SP)                                  ; 0836
                                         7E   D4 0002B       CLRL     -(SP)                                  ;
                              00000000'  EF   9F 0002D       PUSHAB   P.AAA                                  ;
                                         7E   7C 00033       CLRQ     -(SP)                                  ;
                                         7E   D4 00035       CLRL     -(SP)                                  ;
                 00000000G   00              07   FB 00037       CALLS    #7, SYS$GETJPI                         ;
                                         7A   11 0003E       BRB      5$                                     ;
                           63        04  A2   3C 00040 1$:   MOVZWL   4(R2), TPARSE_BLOCK+8                  ; 0839
                      04   A3        08  A2   D0 00044       MOVL     8(R2), TPARSE_BLOCK+12                 ; 0840
                              00000000'  EF   9F 00049       PUSHAB   OWNER_KEYS                             ; 0841
                              00000000'  EF   9F 0004F       PUSHAB   OWNER_STATE                            ;
                                     F8  A3   9F 00055       PUSHAB   TPARSE_BLOCK                           ;
                 00000000G   00              03   FB 00058       CALLS    #3, LIB$TPARSE                         ;
                             0E              50   E8 0005F       BLBS     STATUS, 2$                             ;
                                     04  A2   9F 00062       PUSHAB   4(R2)                                  ; 0846
                                         01   DD 00065       PUSHL    #1                                     ;
                              00000000*  8F   DD 00067       PUSHL    #<<<SET$_FACILITY@16>+4344>+2>         ;
                           64            03   FB 0006D       CALLS    #3, LIB$STOP                           ;
              42 00000000G   00              03   E0 00070 2$:   BBS      #3, SETFILE$FLAGS+2, 5$                ; 0847
                           50 00000000G  00   D0 00078       MOVL     GROUP, R0                              ; 0850
                 000000FF    8F              50   D1 0007F       CMPL     R0, #255                               ;
                                         14   14 00086       BGTR     3$                                     ;
                                         50   D5 00088       TSTL     R0                                     ;
                                         10   19 0008A       BLSS     3$                                     ;
                           51            65   D0 0008C       MOVL     MEMBER, R1                             ; 0852
                 000000FF    8F              51   D1 0008F       CMPL     R1, #255                               ;
                                         04   14 00096       BGTR     3$                                     ;
                                         51   D5 00098       TSTL     R1                                     ;
                                         16   18 0009A       BGEQ     4$                                     ;
                              00000000*  8F   DD 0009C 3$:   PUSHL    #<<<SET$_FACILITY@16>+4584>+2>         ; 0856
                                     04  A2   9F 000A2       PUSHAB   4(R2)                                  ; 0855
                                         01   DD 000A5       PUSHL    #1                                     ;
                              00000000*  8F   DD 000A7       PUSHL    #<<<SET$_FACILITY@16>+4344>+2>         ;
                           64            04   FB 000AD       CALLS    #4, LIB$STOP                           ;
                                         08   11 000B0       BRB      5$                                     ;
              50           50            10   78 000B2 4$:   ASHL     #16, R0, R0                            ; 0857
              66           50            65   C1 000B6       ADDL3    MEMBER, R0, UIC_VALUE                  ;
                           50            01   D0 000BA 5$:   MOVL     #1, R0                                 ; 0861
                                              04 000BD       RET                                             ; 0862

; Routine Size:  190 bytes,    Routine Base:  $CODE$ + 0338
```

```
GLOBAL ROUTINE retent_act (option_block,callback) =
!++
!
! This is the action routine for the /RETENTION qualifier.
! The minimum retention value must be given.  If no maximum retention value is
! specified, a value of twice the minimum (but no more than a week more than
! the minimum) is used.
!
!--
BEGIN

LOCAL
    status,
    temp_desc : BBLOCK[dsc$c_s_bln];

MAP
    option_block : REF BBLOCK;          ! Define the CLI options block

!
! Parse the input, to obtain the minimum and maximum retention times.
!

CH$FILL(0, 8, retmin_value);                ! Zero minimum value
CH$FILL(0, 8, retmax_value);                ! Zero maximum value

tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
IF NOT (status = lib$tparse(tparse_block, ret_state, ret_keys))
THEN
    BEGIN
    SIGNAL(set$_facility^16 + shr$_syntax + sts$k_error,
           1,
           option_block[cli$q_qdvaldesc]);
    RETURN false;                       ! If error in parse, return false
    END;

!
! If a minimum value was not supplied, signal an error
!

IF .retmin_value[0] EQL 0
THEN
    BEGIN
    SIGNAL(set$_facility^16 + shr$_syntax + sts$k_error,
           1,
           option_block[cli$q_qdvaldesc]);
    RETURN false;
    END;

!
! Convert the minimum retention value to 64-bit system delta time format
!
IF NOT (status = LIB$CVT_DTIME(retmin_value, temp_desc))
THEN
    BEGIN
    SIGNAL(set$_facility^16 + shr$_syntax + sts$k_error,
;  946          0920 3              1,
;  947          0921 3              retmin_value,
;  948          0922 3              .status);
;  949          0923 3      RETURN false;
;  950          0924 3      END
;  951          0925 2  ELSE CH$MOVE(8, temp_desc, retmin_value);       ! If no error, put 64-bit
;  952          0926 2                                                  ! delta time in place
;  953          0927 2
;  954          0928 2  !
;  955          0929 2  ! If a maximum value was supplied, then convert it in the same way.
;  956          0930 2  !
;  957          0931 2
;  958          0932 2  IF .retmax_value[0] NEQ 0
;  959          0933 2  THEN
;  960          0934 3      BEGIN
;  961          0935 4      IF NOT (status = LIB$CVT_DTIME(retmax_value, temp_desc))
;  962          0936 3      THEN
;  963          0937 4          BEGIN
;  964          0938 4          SIGNAL(set$_facility^16 + shr$_syntax + sts$k_error,
;  965          0939 4              1,
;  966          0940 4              retmax_value,
;  967          0941 4              .status);
;  968          0942 4          RETURN .status;
;  969          0943 4          END
;  970          0944 3      ELSE CH$MOVE(8, temp_desc, retmax_value);
;  971          0945 3      END
;  972          0946 3
;  973          0947 3  !
;  974          0948 3  ! If no maximum value was supplied, then use twice the minimum value.  If this
;  975          0949 3  ! value is greater than a week, use only a week.
;  976          0950 3  !
;  977          0951 3
;  978          0952 2  ELSE calculate_max(retmin_value, retmax_value);
;  979          0953 2
;  980          0954 2  RETURN true;
;  981          0955 1  END;



                                          OFFC 00000          .ENTRY  RETENT_ACT, Save R2,R3,R4,R5,R6,R7,R8,R9,-   ; 0863
                                                                      R10,R11                                       ;
                        5B 00000000G   00  9E 00002          MOVAB   LIB$CVT_DTIME, R11                            ;
                        5A 00000000G   00  9E 00009          MOVAB   LIB$SIGNAL, R10                               ;
                        59 00000000'   EF  9E 00010          MOVAB   TPARSE_BLOCK+8, R9                            ;
                        58 00000000G   00  9E 00017          MOVAB   RETMIN_VALUE, R8                              ;
                        57 00000000G   00  9E 0001E          MOVAB   RETMAX_VALUE, R7                              ;
                        5E             08  C2 00025          SUBL2   #8, SP                                        ;
        08        00    6E             00  2C 00028          MOVC5   #0, (SP), #0, #8, RETMIN_VALUE                ; 0886
                                       68     0002D
        08        00    6E             00  2C 0002E          MOVC5   #0, (SP), #0, #8, RETMAX_VALUE                ; 0887
                                       67     00033
                        52          04 AC  D0 00034          MOVL    OPTION_BLOCK, R2                              ; 0889
                        69          04 A2  3C 00038          MOVZWL  4(R2), TPARSE_BLOCK+8                         ;
                  04    A9          08 A2  D0 0003C          MOVL    8(R2), TPARSE_BLOCK+12                        ; 0890
                           00000000'   EF  9F 00041          PUSHAB  RET_KEYS                                      ; 0891
```

```
                                                          K 5
SETACT                                                    16-Sep-1984 01:06:01    VAX-11 Bliss-32 V4.0-742          Page  54
V04-000                                                   14-Sep-1984 12:08:59    [CLIUTL.SRC]SETACT.B32;1                (27)

                      00000000'  EF  9F 00047            PUSHAB  RET_STATE                                   ;
                             F8  A9  9F 0004D            PUSHAB  TPARSE_BLOCK                                ;
            00000000G  00        03  FB 00050            CALLS   #3, LIB$TPARSE                              ;
                       56        50  D0 00057            MOVL    R0, STATUS                                  ;
                       04        56  E9 0005A            BLBC    STATUS, 1$                                  ;
                                 68  D5 0005D            TSTL    RETMIN_VALUE                                ; 0904
                                 10  12 0005F            BNEQ    2$                                          ;
                             04  A2  9F 00061 1$:        PUSHAB  4(R2)                                       ; 0909
                                 01  DD 00064            PUSHL   #1                                          ;
                      00000000*  8F  DD 00066            PUSHL   #<<<SET$_FACILITY@16>+4344>+2>              ;
                       6A        03  FB 0006C            CALLS   #3, LIB$SIGNAL                              ;
                                 5B  11 0006F            BRB     7$                                          ; 0910
                           4100  8F  BB 00071 2$:        PUSHR   #^M<R8,SP>                                  ; 0916
                       6B        02  FB 00075            CALLS   #2, LIB$CVT_DTIME                           ;
                       56        50  D0 00078            MOVL    R0, STATUS                                  ;
                       11        56  E8 0007B            BLBS    STATUS, 3$                                  ;
                                 56  DD 0007E            PUSHL   STATUS                                      ; 0922
                                 58  DD 00080            PUSHL   R8                                          ; 0919
                                 01  DD 00082            PUSHL   #1                                          ;
                      00000000*  8F  DD 00084            PUSHL   #<<<SET$_FACILITY@16>+4344>+2>              ;
                       6A        04  FB 0008A            CALLS   #4, LIB$SIGNAL                              ;
                                 3D  11 0008D            BRB     7$                                          ; 0923
        68             6E        08  28 0008F 3$:        MOVC3   #8, TEMP_DESC, RETMIN_VALUE                 ; 0925
                                 67  D5 00093            TSTL    RETMAX_VALUE                                ; 0932
                                 26  13 00095            BEQL    5$                                          ;
                           4080  8F  BB 00097            PUSHR   #^M<R7,SP>                                  ; 0935
                       6B        02  FB 0009B            CALLS   #2, LIB$CVT_DTIME                           ;
                       56        50  D0 0009E            MOVL    R0, STATUS                                  ;
                       13        56  E8 000A1            BLBS    STATUS, 4$                                  ;
                                 56  DD 000A4            PUSHL   STATUS                                      ; 0941
                                 57  DD 000A6            PUSHL   R7                                          ; 0938
                                 01  DD 000A8            PUSHL   #1                                          ;
                      00000000*  8F  DD 000AA            PUSHL   #<<<SET$_FACILITY@16>+4344>+2>              ;
                       6A        04  FB 000B0            CALLS   #4, LIB$SIGNAL                              ;
                       50        56  D0 000B3            MOVL    STATUS, R0                                  ; 0942
                                     04 000B6            RET                                                 ;
        67             6E        08  28 000B7 4$:        MOVC3   #8, TEMP_DESC, RETMAX_VALUE                 ; 0944
                                 0B  11 000BB            BRB     6$                                          ; 0932
                                 57  DD 000BD 5$:        PUSHL   R7                                          ; 0952
                                 58  DD 000BF            PUSHL   R8                                          ;
            00000000G  00        02  FB 000C1            CALLS   #2, CALCULATE_MAX                           ;
                       50        01  D0 000C8 6$:        MOVL    #1, R0                                      ; 0954
                                     04 000CB            RET                                                 ;
                                 50  D4 000CC 7$:        CLRL    R0                                          ; 0955
                                     04 000CE            RET                                                 ;

; Routine Size:  207 bytes,    Routine Base:  $CODE$ + 03F6
```

```
0956   1 ROUTINE test_char =
0957   1 !++
0958   1 !
0959   1 !  This routine is used by TPARSE to process the /RETENTION values.
0960   1 !
0961   1 !  NOTE that this routine references the Argument Pointer (AP) directly,
0962   1 !  due to the fact that TPARSE does not follow the calling standard.
0963   1 !
0964   1 !--
0965   1
0966   2 BEGIN
0967   2
0968   2 BUILTIN AP;                                          ! Declare the argument pointer
0969   2
0970   2 LOCAL
0971   2     ptr,
0972   2     char : BYTE;
0973   2
0974   2 ptr = .ap + $BYTEOFFSET(tpa$b_char);                 ! Look at the character just read
0975   2 char = CH$RCHAR(.ptr);
0976   2
0977   2 IF .char EQL ','
0978   2 THEN RETURN false                                    ! If a comma, then end of string
0979   2 ELSE RETURN true;                                    ! Else continue processing
0980   2
0981   1 END;



                              0000 00000 TEST_CHAR:
                                                   .WORD   Save nothing                         ; 0956
            50        18    AC  9E 00002           MOVAB   24(AP), PTR                          ; 0974
            51              60  90 00006           MOVB    (PTR), CHAR                          ; 0975
            2C              51  91 00009           CMPB    CHAR, #44                            ; 0977
                            03  12 0000C           BNEQ    1$                                   ;
                            50  D4 0000E           CLRL    R0                                   ; 0979
                                04 00010           RET                                          ;
            50              01  D0 00011 1$:       MOVL    #1, R0                               ;
                                04 00014           RET                                          ; 0981

; Routine Size:  21 bytes,     Routine Base:  $CODE$ + 04C5
```

```
     1 GLOBAL ROUTINE user_act (option_block, callback) =
     1 !++
     1 !
     1 ! This is the action routine for the USER_NAME qualifier of SET VOLUME.  It
     1 ! retrieves the value of the string, checks that it is no longer than
     1 ! twelve characters, and stores a descriptor pointing to it.
     1 !
     1 !--
     2 BEGIN
     2
     2 OWN user_label : VECTOR[12,BYTE];        ! Place to put process username
     2
     2 LOCAL status;                            ! Status return
     2
     2 MAP option_block : REF BBLOCK;           ! Define the cli block
     2
     2 !
     2 ! If no username was specified, use the current process username.
     2 !
     2 IF .option_block[cli$w_qdvalsiz] EQL 0
     2 THEN
     3     BEGIN
P    3     $GETJPI(ITMLST = UPLIT(WORD(4, jpi$_username),       ! Get the username
P    3                            user_label,                   ! Store it here
P    3                            user_value[0],                ! Store length here
     3                            0));
     3     user_value[1] = user_label;
     3     END
     3
     2 ELSE
     3     BEGIN
     3
     3 !
     3 ! Check that the string is no longer than twelve characters.
     3 !
     3     IF .option_block[cli$w_qdvalsiz] GTR 12
     3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
     3                      1,
     3                      option_block[cli$q_qdvaldesc],
     3                      set$_facility^16 + shr$_valerr + sts$k_error);
     3 !
     3 ! Record the length and location in USER_VALUE.
     3 !
     3     user_value[0] = .option_block[cli$w_qdvalsiz];
     3     user_value[1] = .option_block[cli$a_qdvaladr];
     2     END;
     2
     2 RETURN true;
     1 END;


                                                           .PSECT  $PLIT$,NOWRT,NOEXE,2

                          0202  0004  00010 P.AAB:         .WORD   4, 514                          ;
                 00000000G 00000000'  00014                .ADDRESS USER_LABEL, USER_VALUE         ;
                           00000000   0001C                .LONG   0                               ;
```

```
                                                     .PSECT  $OWN$,NOEXE,2

                                     0002C USER_LABEL:
                                                     .BLKB   12


                                                     .PSECT  $CODE$,NOWRT,2

                                 000C 00000          .ENTRY  USER_ACT, Save R2,R3                      ; 0982
                    53 00000000G   00   9E 00002          MOVAB   USER_VALUE+4, R3                          ;
                    52         04   AC   D0 00009          MOVL    OPTION_BLOCK, R2                          ; 1001
                               04   A2   B5 0000D          TSTW    4(R2)                                     ;
                                    1E   12 00010          BNEQ    1$                                        ;
                                    7E   7C 00012          CLRQ    -(SP)                                     ; 1007
                                    7E   D4 00014          CLRL    -(SP)                                     ;
                       00000000'    EF   9F 00016          PUSHAB  P.AAB                                     ;
                                    7E   7C 0001C          CLRQ    -(SP)                                     ;
                                    7E   D4 0001E          CLRL    -(SP)                                     ;
             00000000G  00          07   FB 00020          CALLS   #7, SYS$GETJPI                            ;
                    63 00000000'    EF   9E 00027          MOVAB   USER_LABEL, USER_VALUE+4                  ; 1008
                                    27   11 0002E          BRB     3$                                        ; 1001
                    0C         04   A2   B1 00030 1$:      CMPW    4(R2), #12                                ; 1017
                                    18   1B 00034          BLEQU   2$                                        ;
                       00000000*    8F   DD 00036          PUSHL   #<<<SET$_FACILITY@16>+4584>+2>            ; 1021
                               04   A2   9F 0003C          PUSHAB  4(R2)                                     ; 1020
                                    01   DD 0003F          PUSHL   #1                                        ;
                       00000000*    8F   DD 00041          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>            ;
             00000000G  00          04   FB 00047          CALLS   #4, LIB$STOP                              ;
                   FC   A3     04   A2   3C 0004E 2$:      MOVZWL  4(R2), USER_VALUE                         ; 1025
                        63     08   A2   D0 00053          MOVL    8(R2), USER_VALUE+4                       ; 1026
                        50          01   D0 00057 3$:      MOVL    #1, R0                                    ; 1029
                                         04 0005A          RET                                               ; 1030

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 04DA
; 1060        1031  1 GLOBAL ROUTINE vprot_act (option_block, callback) =
; 1061        1032  1 !++
; 1062        1033  1 !
; 1063        1034  1 !  This is the action routine for the PROTECTION qualifier of SET VOLUME.
; 1064        1035  1 !  The protection is parsed and stored.
; 1065        1036  1 !
; 1066        1037  1 !--
; 1067        1038  2 BEGIN
; 1068        1039  2
; 1069        1040  2 LOCAL
; 1070        1041  2     status,                         ! Status return
; 1071        1042  2     temp;                           ! Temporary place for FPROT_VALUE
; 1072        1043  2
; 1073        1044  2 MAP option_block : REF BBLOCK;    ! Define CLI block
; 1074        1045  2
; 1075        1046  2 !
; 1076        1047  2 !  Stuff the TPARSE block with the string
; 1077        1048  2 !
; 1078        1049  2 tparse_block[tpa$l_stringcnt] = .option_block[cli$w_qdvalsiz];
; 1079        1050  2 tparse_block[tpa$l_stringptr] = .option_block[cli$a_qdvaladr];
; 1080        1051  2
; 1081        1052  2 temp = .fprot_value;                              ! Save contents of FPROT
; 1082        1053  2 fprot_value = 0;                                  ! Initialize file protection value
; 1083        1054  2
; 1084        1055  2 !
; 1085        1056  2 !  Now to parse the protection given.  When finished, FPROT_VALUE will
; 1086        1057  2 !  have the following values:
; 1087        1058  2 !
; 1088        1059  2 !  FPROT_VALUE[low_word] = protection value
; 1089        1060  2 !  FPROT_VALUE[high_word] = group mask i.e. SYSTEM, OWNER, GROUP, WORLD
; 1090        1061  2 !
; 1091        1062  3 IF NOT (status = LIB$TPARSE(tparse_block,
; 1092        1063  3                              pro_state,
; 1093        1064  3                              pro_keys))
; 1094        1065  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
; 1095        1066  2                  1,
; 1096        1067  2                  option_block[cli$q_qdvaldesc],
; 1097        1068  2                  .status);
; 1098        1069  2
; 1099        1070  2 vprot_value = .fprot_value;                       ! Store VPROT value
; 1100        1071  2 fprot_value = .temp;                              ! Restore FPROT value
; 1101        1072  2
; 1102        1073  2 RETURN true;
; 1103        1074  1 END;



                                   003C 00000        .ENTRY  VPROT_ACT, Save R2,R3,R4,R5             ; 1031
              55 00000000'   EF    9E 00002        MOVAB   TPARSE_BLOCK+8, R5                       ;
              54 00000000G   00    9E 00009        MOVAB   FPROT_VALUE, R4                          ;
              52        04   AC    D0 00010        MOVL    OPTION_BLOCK, R2                         ; 1049
              65        04   A2    3C 00014        MOVZWL  4(R2), TPARSE_BLOCK+8                    ;
         04   A5        08   A2    D0 00018        MOVL    8(R2), TPARSE_BLOCK+12                   ; 1050
              53              64   D0 0001D        MOVL    FPROT_VALUE, TEMP                        ; 1052
                              64   D4 00020        CLRL    FPROT_VALUE                              ; 1053
```

```
                         00000000'  EF  9F 00022          PUSHAB  PRO_KEYS                              ; 1062
                         00000000'  EF  9F 00028          PUSHAB  PRO_STATE                             ;
                                F8  A5  9F 0002E          PUSHAB  TPARSE_BLOCK                          ;
          00000000G  00             03  FB 00031          CALLS   #3, LIB$TPARSE                        ;
                     14             50  E8 00038          BLBS    STATUS, 1$                            ;
                                    50  DD 0003B          PUSHL   STATUS                                ; 1068
                                04  A2  9F 0003D          PUSHAB  4(R2)                                 ; 1067
                                    01  DD 00040          PUSHL   #1                                    ;
                         00000000*  8F  DD 00042          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>        ;
          00000000G  00             04  FB 00048          CALLS   #4, LIB$STOP                          ;
          00000000G  00             64  D0 0004F  1$:     MOVL    FPROT_VALUE, VPROT_VALUE              ; 1070
                     64             53  D0 00056          MOVL    TEMP, FPROT_VALUE                     ; 1071
                     50             01  D0 00059          MOVL    #1, R0                                ; 1073
                                        04 0005C          RET                                           ; 1074

; Routine Size:  93 bytes,    Routine Base:  $CODE$ + 0535
; 1105          1075   1 GLOBAL ROUTINE vrsn_act (option_block,callback) =
; 1106          1076   1 !++
; 1107          1077   1 !
; 1108          1078   1 ! This is the action routine for the VERSION_LIMIT qualifier.  The value of
; 1109          1079   1 ! the version limit is collected.
; 1110          1080   1 !
; 1111          1081   1 !--
; 1112          1082   2 BEGIN
; 1113          1083   2
; 1114          1084   2 LOCAL
; 1115          1085   2     status,
; 1116          1086   2     desc : BBLOCK[dsc$c_s_bln];
; 1117          1087   2
; 1118          1088   2 MAP
; 1119          1089   2     option_block : REF BBLOCK;          ! Define the CLI options block
; 1120          1090   2
; 1121          1091   2 vrsn_value = 32767;                     ! Preset to no limit
; 1122          1092   2
; 1123          1093   2 !
; 1124          1094   2 ! See if a value was present.  If yes, use it.  Otherwise, use default
; 1125          1095   2 !
; 1126          1096   2 IF .option_block[cli$w_qdvalsiz] EQL 0
; 1127          1097   2 THEN RETURN true;
; 1128          1098   2
; 1129          1099   3 IF NOT (status = LIB$CVT_DTB(.option_block[cli$w_qdvalsiz],
; 1130          1100   3                              .option_block[cli$a_qdvaladr],
; 1131          1101   3                              vrsn_value))
; 1132          1102   2 THEN
; 1133          1103   3     BEGIN
; 1134          1104   3     SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
; 1135          1105   3                 1,
; 1136          1106   3                 option_block[cli$q_qdvaldesc],
; 1137          1107   3                 .status);
; 1138          1108   3     END
; 1139          1109   2 ELSE
; 1140          1110   3     BEGIN
; 1141          1111   4     IF NOT (.vrsn_value GEQ 0 AND .vrsn_value LEQ 65535)
; 1142          1112   3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,
; 1143          1113   3                      1,
; 1144          1114   3                      option_block[cli$q_qdvaldesc],
; 1145          1115   3                      set$_facility^16 + shr$_valerr + sts$k_error);
; 1146          1116   2     END;
; 1147          1117   2
; 1148          1118   2 RETURN true;
; 1149          1119   1 END;



                                         000C 00000          .ENTRY  VRSN_ACT, Save R2,R3                                ; 1075
                      53 00000000G   00   9E 00002          MOVAB   VRSN_VALUE, R3                                      ;
                      5E             08   C2 00009          SUBL2   #8, SP                                              ;
                      63      7FFF   8F   3C 0000C          MOVZWL  #32767, VRSN_VALUE                                  ; 1091
                      52        04   AC   D0 00011          MOVL    OPTION_BLOCK, R2                                    ; 1096
                                04   A2   B5 00015          TSTW    4(R2)                                               ;
                                     3D   13 00018          BEQL    4$                                                  ;
```

```
                                   53  DD 0001A        PUSHL   R3                                             ; 1099
                         08        A2  DD 0001C        PUSHL   8(R2)                                          ; 1100
                  7E     04        A2  3C 0001F        MOVZWL  4(R2), -(SP)                                   ; 1099
      00000000G   00               03  FB 00023        CALLS   #3, LIB$CVT_DTB                                ;
                  04               50  E8 0002A        BLBS    STATUS, 1$                                     ;
                                   50  DD 0002D        PUSHL   STATUS                                         ; 1107
                                   14  11 0002F        BRB     3$                                             ; 1106
                  50               63  D0 00031 1$:    MOVL    VRSN_VALUE, R0                                 ; 1111
                                   09  19 00034        BLSS    2$                                             ;
      0000FFFF    8F               50  D1 00036        CMPL    R0, #65535                                     ;
                                   18  15 0003D        BLEQ    4$                                             ;
                     00000000*     8F  DD 0003F 2$:    PUSHL   #<<<SET$_FACILITY@16>+4584>+2>                 ; 1115
                         04        A2  9F 00045 3$:    PUSHAB  4(R2)                                          ; 1114
                                   01  DD 00048        PUSHL   #1                                             ;
                     00000000*     8F  DD 0004A        PUSHL   #<<<SET$_FACILITY@16>+4344>+2>                 ;
      00000000G   00               04  FB 00050        CALLS   #4, LIB$STOP                                   ;
                  50               01  D0 00057 4$:    MOVL    #1, R0                                         ; 1118
                                       04 0005A        RET                                                    ; 1119

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 0592
; 1151          1120  1 GLOBAL ROUTINE window_act (option_block, callback) =
; 1152          1121  1 !++
; 1153          1122  1 !
; 1154          1123  1 ! This is the action routine for the /WINDOWS qualifier.  It retrieves the
; 1155          1124  1 ! value and performs bounds checking on it.
; 1156          1125  1 !
; 1157          1126  1 !--
; 1158          1127  2 BEGIN
; 1159          1128  2
; 1160          1129  2 LOCAL
; 1161          1130  2     status,                     ! Status return
; 1162          1131  2     desc : BBLOCK[dsc$c_s_bln]; ! General descriptor
; 1163          1132  2
; 1164          1133  2 MAP option_block : REF BBLOCK;  ! Define the CLI block
; 1165          1134  2
; 1166          1135  2 window_value = 7;                         ! Set up the default
; 1167          1136  2
; 1168          1137  2 !
; 1169          1138  2 ! If a value was specified, use it; otherwise, use the default.
; 1170          1139  2 !
; 1171          1140  2 IF .option_block[cli$w_qdvalsiz] EQL 0
; 1172          1141  2 THEN RETURN true;
; 1173          1142  2
; 1174          1143  2 !
; 1175          1144  2 ! Convert the value
; 1176          1145  2 !
; 1177          1146  3 IF NOT (status = LIB$CVT_DTB(.option_block[cli$w_qdvalsiz],
; 1178          1147  3                             .option_block[cli$a_qdvaladr],
; 1179          1148  3                             window_value))
; 1180          1149  2 THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,  ! Signal a syntax error
; 1181          1150  2                 1,
; 1182          1151  2                 option_block[cli$q_qdvaldesc],
; 1183          1152  2                 .status)
; 1184          1153  2 ELSE
; 1185          1154  3     BEGIN
; 1186          1155  4     IF NOT (.window_value GEQ 7          ! Check that value is in range
; 1187          1156  4                 AND
; 1188          1157  4             .window_value LEQ 80)
; 1189          1158  3     THEN SIGNAL_STOP(set$_facility^16 + shr$_syntax + sts$k_error,       ! If not, exit with an error.
; 1190          1159  3                     1,
; 1191          1160  3                     option_block[cli$q_qdvaldesc],
; 1192          1161  3                     set$_facility^16 + shr$_valerr + sts$k_error);
; 1193          1162  2     END;
; 1194          1163  2 RETURN true;
; 1195          1164  1 END;




                              000C 00000          .ENTRY  WINDOW_ACT, Save R2,R3                    ; 1120
          53 00000000G   00    9E 00002          MOVAB   WINDOW_VALUE, R3                          ;
          5E             08    C2 00009          SUBL2   #8, SP                                    ;
          63             07    D0 0000C          MOVL    #7, WINDOW_VALUE                          ; 1135
          52       04    AC    D0 0000F          MOVL    OPTION_BLOCK, R2                          ; 1140
                   04    A2    B5 00013          TSTW    4(R2)                                     ;
                         40    13 00016          BEQL    4$                                        ;
```

```
                                   53  DD 00018          PUSHL   R3                                        ; 1146
                           08      A2  DD 0001A          PUSHL   8(R2)                                     ; 1147
                 7E        04      A2  3C 0001D          MOVZWL  4(R2), -(SP)                              ; 1146
       00000000G 00                03  FB 00021          CALLS   #3, LIB$CVT_DTB                           ;
                 04                50  E8 00028          BLBS    STATUS, 1$                                ;
                                   50  DD 0002B          PUSHL   STATUS                                    ; 1152
                                   17  11 0002D          BRB     3$                                        ; 1151
                 50                63  D0 0002F 1$:      MOVL    WINDOW_VALUE, R0                          ; 1155
                 07                50  D1 00032          CMPL    R0, #7                                    ;
                                   09  19 00035          BLSS    2$                                        ;
       00000050  8F                50  D1 00037          CMPL    R0, #80                                   ; 1157
                                   18  15 0003E          BLEQ    4$                                        ;
                        00000000*  8F  DD 00040 2$:      PUSHL   #<<<SET$_FACILITY@16>+4584>+2>            ; 1161
                           04      A2  9F 00046 3$:      PUSHAB  4(R2)                                     ; 1160
                                   01  DD 00049          PUSHL   #1                                        ;
                        00000000*  8F  DD 0004B          PUSHL   #<<<SET$_FACILITY@16>+4344>+2>            ;
       00000000G 00                04  FB 00051          CALLS   #4, LIB$STOP                              ;
                 50                01  D0 00058 4$:      MOVL    #1, R0                                    ; 1163
                                       04 0005B          RET                                               ; 1164

; Routine Size:  92 bytes,    Routine Base:  $CODE$ + 05ED
; 1197          1165  1 END
; 1198          1166  0 ELUDOM


                                                                .EXTRN  LIB$SIGNAL, LIB$STOP

;                                   PSECT SUMMARY
;
;       Name                      Bytes                   Attributes
;
;    $OWN$                            56  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    _LIB$KEY0$                       40  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;    _LIB$STATE$                     642  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;    _LIB$KEY1$                      104  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;    $CODE$                         1609  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    .  ABS  .                         0  NOVEC,NOWRT,NORD ,NOEXE,NOSHR,  LCL,  ABS,  CON,NOPIC,ALIGN(0)
;    $PLIT$                           32  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                              Library Statistics
;
;                                              -------- Symbols --------       Pages       Processing
;       File                                   Total    Loaded   Percent       Mapped      Time
;
;    _$255$DUA28:[SYSLIB]LIB.L32;1            18619        30         0         1000        00:01.8
;    _$255$DUA28:[SYSLIB]CLIMAC.L32;1            14         0         0            9        00:00.1
;    _$255$DUA28:[SYSLIB]TPAMAC.L32;1            42        29        69           14        00:00.2



;                                   COMMAND QUALIFIERS

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

; Size:          1609 code + 874 data bytes
; Run Time:         01:06.7
; Elapsed Time:     03:44.0
; Lines/CPU Min:     1049
; Lexemes/CPU-Min: 69494
; Memory Used:  274 pages
; Compilation Complete
```

0052 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SETDEVICE
LIS

SETDEVS
LIS

SETCLUSUB
LIS

SETDIR
LIS

SETACT
LIS

SETAUDIT
LIS

SETCLUSTR
LIS