```
   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
```

```
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      000000
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  PPPPPPPP    RRRRRRRR      000000
SS          EE              TT      PP      PP  RR      RR  00      00
SS          EE              TT      PP      PP  RR      RR  00      00
SS          EE              TT      PP      PP  RR      RR  00      00
SS          EE              TT      PP      PP  RR      RR  00      00
  SSSSSS    EEEEEEEE        TT      PPPPPPPP    RRRRRRRR    00      00
  SSSSSS    EEEEEEEE        TT      PPPPPPPP    RRRRRRRR    00      00
        SS  EE              TT      PP          RR  RR      00      00
        SS  EE              TT      PP          RR  RR      00      00
        SS  EE              TT      PP          RR    RR    00      00      ....
        SS  EE              TT      PP          RR    RR    00      00      ....
SSSSSSSS    EEEEEEEEEE      TT      PP          RR      RR    000000        ....
SSSSSSSS    EEEEEEEEEE      TT      PP          RR      RR    000000        ....


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
```

```
;     1          0001  0 MODULE setpro ( IDENT = 'V04-000',
;     2          0002  0                  ADDRESSING_MODE (EXTERNAL = GENERAL)) =
;     3          0003  1 BEGIN
;     4          0004  1
;     5          0005  1 !
;     6          0006  1 !*****************************************************************************
;     7          0007  1 !*                                                                           *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                     *
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    17          0017  1 !*  TRANSFERRED.                                                              *
;    18          0018  1 !*                                                                           *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    21          0021  1 !*  CORPORATION.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*****************************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 ! FACILITY:  SETPRO Command
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This utility sets protection for files and directories.
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT:
;    37          0037  1 !
;    38          0038  1 !       VAX/VMS operating system. Privileged user mode,
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:  Greg Robert, Nov 1979
;    41          0041  1 !
;    42          0042  1 ! Modified by:
;    43          0043  1 !
;    44          0044  1 !       V03-006 GAS0112         Gerry Smith             29-Mar-1983
;    45          0045  1 !               Remove references to old command dispatcher.
;    46          0046  1 !
;    47          0047  1 !       V03-005 GAS0100         Gerry Smith             12-Mar-1983
;    48          0048  1 !               Remove all references to old CLI interface.
;    49          0049  1 !
;    50          0050  1 !       V03-004 GAS0094         Gerry Smith             19-Nov-1982
;    51          0051  1 !               Fix /LOG qualifier.
;    52          0052  1 !
;    53          0053  1 !       V03-003 GAS0093         Gerry Smith             27-Oct-1982
;    54          0054  1 !               Fix protection parse algorithm.  Also, change the
;    55          0055  1 !               protection descriptor to a dynamic one, and let the
;    56          0056  1 !               system de/allocate memory as appropriate.
;    57          0057  1 !
```

```
;    58          0058  1 !       V03-002 GAS0092         Gerry Smith             22-Oct-1982
;    59          0059  1 !               Hybridize the CLI interface, so that if /DEVICE is
;    60          0060  1 !               not specified, use the new interface.  This fixes a
;    61          0061  1 !               problem that keeps SET PROT=prot from working correctly.
;    62          0062  1 !
;    63          0063  1 !       V03-001 GAS0091         Gerry Smith             19-Oct-1982
;    64          0064  1 !               Change input request for new CLD syntax.
;    65          0065  1 !
;    66          0066  1 !       V02-006 MLJ0066         Martin L. Jack, 29-Dec-1981  13:36
;    67          0067  1 !               Integrate new LIB$SET_FILE_PROT (formerly LIB$SET_PROT).
;    68          0068  1 !
;    69          0069  1 !       V205    GRR2005         G. R. Robert            17-Nov-1981
;    70          0070  1 !               Fixed handling of filespecs in confirmation and logging
;    71          0071  1 !               messages to use expanded string if resultant not available,
;    72          0072  1 !               and to use RMS parse to insure that passwords are blanked.
;    73          0073  1 !
;    74          0074  1 !       V204    GRR2004         G. R. Robert            16-Nov-1981
;    75          0075  1 !               Made all external references addressing mode general.
;    76          0076  1 !
;    77          0077  1 !       V203    GRR2003         G. R. Robert            14-Sep-1981
;    78          0078  1 !               Fixed improper descriptor initialization in confirm_action
;    79          0079  1 !               by making them OWN instead of LOCAL.
;    80          0080  1 !
;    81          0081  1 !       002     GRR2002         Greg Robert     12-Jun-1981
;    82          0082  1 !               Added non-fatal error message to continue
;    83          0083  1 !               processing files after a fatal error.
;    84          0084  1 !
;    85          0085  1 !       001     TMH0001         Tim Halvorsen   21-Jan-1980
;    86          0086  1 !               Add dispatching for SET PROT/DEVICE to SETDEV and include
;    87          0087  1 !               code from require file into here and obsolete require file.
;    88          0088  1 !--
;    89          0089  1
;    90          0090  1 !
;    91          0091  1 ! Include files
;    92          0092  1 !
;    93          0093  1
;    94          0094  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';      ! VAX/VMS common definitions
```

```
;    96          0095  1 !
;    97          0096  1 ! Table of contents
;    98          0097  1 !
;    99          0098  1
;   100          0099  1 FORWARD ROUTINE
;   101          0100  1     set$protection : NOVALUE,          ! Main setpro routine
;   102          0101  1     setpro_action,                     ! Called for each file
;   103          0102  1     get_output_file,                   ! Get next output file
;   104          0103  1     confirm_action,                    ! Interrogate user
;   105          0104  1     log_results,                       ! Inform user of results
;   106          0105  1     expand_prot,                       ! Converts binary prot to ascii
;   107          0106  1     parse_class,                       ! Parse protection of one user class
;   108          0107  1     search_error,                      ! LIB$FILE_SCAN error handler
;   109          0108  1     cli_error;                         ! Handler for CLI errors
;   110          0109  1
;   111          0110  1
;   112          0111  1 !
;   113          0112  1 ! External routines
;   114          0113  1 !
;   115          0114  1
;   116          0115  1 EXTERNAL ROUTINE
;   117          0116  1     cli$get_value,                     ! Get value from CLI interface
;   118          0117  1     cli$present,                       ! Test for presence of qualifier
;   119          0118  1     sys$fao,                           ! Formats ascii output
;   120          0119  1     sys$setprv,                        ! Set privileges
;   121          0120  1     lib$get_command,                   ! Reads from SYS$COMMAND
;   122          0121  1     lib$file_scan,                     ! Handles stickiness etc
;   123          0122  1     lib$set_file_prot,                 ! Set file protection
;   124          0123  1     set$device;                        ! SET PROT/DEVICE action routine
;   125          0124  1
;   126          0125  1 !
;   127          0126  1 ! External data references
;   128          0127  1 !
;   129          0128  1
;   130          0129  1 EXTERNAL LITERAL
;   131          0130  1     set$_pronotchg;                    ! Error message - "Protection not changed"
```

```
;  133          0131  1
;  134          0132  1 !
;  135          0133  1 !       DEFINE VMS BLOCK STRUCTURES
;  136          0134  1 !
;  137          0135  1
;  138          0136  1 STRUCTURE
;  139          0137  1         BBLOCK [O, P, S, E; N] =
;  140          0138  1                [N]
;  141          0139  1                 (BBLOCK+O)<P,S,E>;
```

```
;  143          C140  1 MACRO
;  144          0141  1
;  145          0142  1 !
;  146          0143  1 ! A) Macro to describe a string
;  147          0144  1 ! B) Macro to generate a quadword string descriptor
;  148          0145  1 ! C) Macro to generate the address of a string descriptor
;  149          0146  1
;  150          0147  1     PRIMDESC (str) = %CHARCOUNT (str), UPLIT (%ASCII str)%,
;  151          0148  1     INITDESC (str) = BBLOCK [DSC$C_S_BLN] INITIAL (PRIMDESC (str))%,
;  152          0149  1     ADDRDESC (str) = UPLIT (PRIMDESC (str))%,
;  153          0150  1
;  154          0151  1 !
;  155          0152  1 !       Macro to signal a condition to the handler
;  156          0153  1 !
;  157          0154  1
;  158        M 0155  1     write_message(msg) =
;  159        M 0156  1         SIGNAL(msg                          ! Pass the message code
;  160        M 0157  1         %IF %LENGTH GTR 1                   ! and if more than 1 arg
;  161          0158  1             %THEN ,%REMAINING %FI) %;       ! then the rest too
;  162          0159  1
```

```
;  164          0160  1 !
;  165          0161  1 !       Define message codes.
;  166          0162  1 !
;  167          0163  1
;  168        P 0164  1 $shr_msgdef(set,119,local,
;  169        P 0165  1           (protected,info),       ! File protection changed
;  170        P 0166  1           (syntax,severe),        ! Failure in parse routines
;  171          0167  1           (searchfail,error));    ! Could not find file
;  172          0168  1
```

```
;  174          0169  1 !
;  175          0170  1 !       EQUATED SYMBOLS
;  176          0171  1 !
;  177          0172  1
;  178          0173  1 LITERAL
;  179          0174  1     TRUE        = 1,                    ! BOOLEAN TRUE
;  180          0175  1     FALSE       = 0,                    ! BOOLEAN FALSE
;  181          0176  1     OK          = 1,                    ! SUCCESS RETURN CODE
;  182          0177  1     ERROR       = 2;                    ! ERROR RETURN CODE
;  183          0178  1
;  184          0179  1
;  185          0180  1
;  186          0181  1 !
;  187          0182  1 !       Define CLI qualifier bit flag numbers (see setpro$flags)
;  188          0183  1 !
;  189          0184  1
;  190          0185  1 LITERAL
;  191        P 0186  1     $EQULST (QUAL_,,0,1,
;  192        P 0187  1         (protection,),          ! /PROTECTION
;  193        P 0188  1         (log,),                 ! /LOG
;  194        P 0189  1         (confirm,),             ! /CONFIRM
;  195          0190  1         (device,));             ! /DEVICE
```

```
;  197          0191  1 !
;  198          0192  1 ! Storage definitions
;  199          0193  1 !
;  200          0194  1
;  201          0195  1 OWN
;  202          0196  1     setpro$flags        : BITVECTOR[32] ! General DCL flagword
;  203          0197  1         INITIAL(0),                     ! Initially none present
;  204          0198  1
;  205          0199  1     qualifier$flags     : BITVECTOR[32] ! Qualifier presence bitmap
;  206          0200  1         INITIAL(0),                     ! Initially clear
;  207          0201  1
;  208          0202  1     setpro_prot         : WORD          ! Contains /PROTECTION value
;  209          0203  1                           INITIAL(0),   ! In case user gives no value
;  210          0204  1
;  211          0205  1     setpro_mask         : WORD          ! Contains /PROTECTION mask
;  212          0206  1                           INITIAL(0),   ! In case user gives no value
;  213          0207  1
;  214          0208  1     global_mask         : WORD,         ! Command wide mask
;  215          0209  1     global_prot         : WORD,         ! Command wide protection
;  216          0210  1
;  217          0211  1     result_mask         : WORD,         ! Combined mask
;  218          0212  1     result_prot         : WORD,         ! Combined protection
;  219          0213  1
;  220          0214  1     oldpriv             : BBLOCK[8],    ! Permanent priv's stored here
;  221          0215  1
;  222          0216  1     newpriv             : BBLOCK[8]     ! Mask describing system priv
;  223          0217  1                   PRESET ([prv$v_sysprv]=true), ! Initialize this bit
;  224          0218  1
;  225          0219  1     worst_error: BBLOCK[4]              ! Worst error encountered
;  226          0220  1                  INITIAL(ss$_normal);   ! Initially normal status
```

```
;  228          0221  1 !
;  229          0222  1 ! Define RMS blocks
;  230          0223  1 !
;  231          0224  1
;  232          0225  1 OWN
;  233          0226  1     output_filename:                    ! Place to store the file name
;  234          0227  1                 VECTOR[nam$c_maxrss,BYTE],
;  235          0228  1
;  236          0229  1     output_nam_result:                  ! Resultant output name
;  237          0230  1                 VECTOR [nam$c_maxrss,BYTE],
;  238          0231  1
;  239          0232  1     output_nam_expanded:                ! Expanded output name
;  240          0233  1                 VECTOR [nam$c_maxrss,BYTE],
;  241          0234  1
;  242        P 0235  1     output_rlf: $NAM(                   ! Related name block
;  243          0236  1                 ),
;  244          0237  1
;  245        P 0238  1     output_nam: $NAM(                   ! File name block
;  246        P 0239  1         RLF = output_rlf,               ! Point to related name block
;  247        P 0240  1         ESA = output_nam_expanded,      ! File name before open
;  248        P 0241  1         ESS = nam$c_maxrss,
;  249        P 0242  1         RSA = output_nam_result,        ! File name after open
;  250          0243  1         RSS = nam$c_maxrss),
;  251          0244  1
;  252        P 0245  1     output_fab: $FAB(                   ! FAB for output
;  253          0246  1         NAM = output_nam);              ! Address of name block
;  254          0247  1
```

```
;  256          0248  1 GLOBAL ROUTINE set$protection : NOVALUE =
;  257          0249  1
;  258          0250  1 !++
;  259          0251  1 ! Functional description
;  260          0252  1 !
;  261          0253  1 !       This is the main control routine for the SET PROTECTION cmd.
;  262          0254  1 !       It is called from the command language interpreter to
;  263          0255  1 !       set protection for files and directories.
;  264          0256  1 !
;  265          0257  1 ! Calling sequence
;  266          0258  1 !
;  267          0259  1 !       set$protection (option_desc_block) from the CLI as an
;  268          0260  1 !       action routine for the /PROTECTION OPTION of the SET command
;  269          0261  1 !
;  270          0262  1 ! Input parameters
;  271          0263  1 !
;  272          0264  1 !       None
;  273          0265  1 !
;  274          0266  1 ! Output parameters
;  275          0267  1 !
;  276          0268  1 !       None
;  277          0269  1 !
;  278          0270  1 ! Routine value
;  279          0271  1 !
;  280          0272  1 !       Worst error encountered during processing or SS$_NORMAL.
;  281          0273  1 !
;  282          0274  1 !----
;  283          0275  1
;  284          0276  2 BEGIN
;  285          0277  2
;  286          0278  2 LOCAL
;  287          0279  2     prot_desc : $BBLOCK[dsc$c_s_bln],   ! Protection descriptor
;  288          0280  2     status;                             ! Status return
;  289          0281  2
;  290          0282  2
;  291          0283  2 !
;  292          0284  2 ! The SET command is installed with system privileges (SYSPRV), however
;  293          0285  2 ! we don't want the user to have this much power unless s/he
;  294          0286  2 ! explicitly asks for it.  Therefore we will inspect the existing
;  295          0287  2 ! privileges.  If SYSPRV is true on a process permanent basis we will
;  296          0288  2 ! leave it alone, but if only enabled on a temporary basis (for this
;  297          0289  2 ! image) then we will disable the privilege
;  298          0290  2 !
;  299          0291  2
;  300          0292  2 !
;  301          0293  2 ! First we get the permanent privileges
;  302          0294  2 !
;  303          0295  2
;  304          0296  3 IF NOT (status =
;  305          0297  3 SYS$SETPRV      (1,             ! Enable
;  306          0298  3                 0,              ! No mask supplied
;  307          0299  3                 1,              ! Permanent privileges
;  308          0300  3                 oldpriv)        ! Place to deposit current privileges
;  309          0301  3 ) THEN BEGIN
;  310          0302  3         write_message (.status);         ! Tell the user whats wrong
;  311          0303  3         return;                          ! And give up
;  312          0304  2         END;
```

```
;  313          0305  2
;  314          0306  2 !
;  315          0307  2 ! Now decide whether or not to clear the privilege
;  316          0308  2 !
;  317          0309  2
;  318          0310  3 IF NOT (.oldpriv[prv$v_sysprv]) ! If privilege NOT permanent
;  319          0311  3 THEN BEGIN
;  320          0312  4         IF NOT (status =
;  321          0313  4         SYS$SETPRV      (0,     ! Disable
;  322          0314  4                         newpriv,! This mask
;  323          0315  4                         0,      ! Temporary
;  324          0316  4                         0)      ! Don't return old mask
;  325          0317  4         ) THEN BEGIN
;  326          0318  4                 write_message (.status);! Tell the user what's wrong
;  327          0319  4                 return;         ! And give up
;  328          0320  4             END
;  329          0321  2     END;
;  330          0322  2
```

```
;  332          0323  2 !
;  333          0324  2 !       If SET PROT/DEVICE, call SETDEV module to do the job
;  334          0325  2 !
;  335          0326  2 IF cli$present(%ASCID 'DEVICE')
;  336          0327  2 THEN
;  337          0328  3     BEGIN
;  338          0329  3     set$device(1);
;  339          0330  3     RETURN;
;  340          0331  2     END;
;  341          0332  2
;  342          0333  2 !
;  343          0334  2 !       Parse the SET PROTECTION= value if any
;  344          0335  2 !
;  345          0336  2 $init_dyndesc(prot_desc);               ! Make a dynamic descriptor
;  346          0337  2
;  347          0338  2 IF cli$present(%ASCID 'OPTION.PROTECTION.SYSTEM')
;  348          0339  2 THEN
;  349          0340  3     BEGIN
;  350          0341  3     setpro_mask = .setpro_mask OR %X'000F';
;  351          0342  3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.SYSTEM',prot_desc)
;  352          0343  3     THEN setpro_prot = parse_class(prot_desc);
;  353          0344  2     END;
;  354          0345  2 IF cli$present(%ASCID 'OPTION.PROTECTION.OWNER')
;  355          0346  2 THEN
;  356          0347  3     BEGIN
;  357          0348  3     setpro_mask = .setpro_mask OR %X'00F0';
;  358          0349  3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.OWNER',prot_desc)
;  359          0350  3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^4;
;  360          0351  2     END;
;  361          0352  2 IF cli$present(%ASCID 'OPTION.PROTECTION.GROUP')
;  362          0353  2 THEN
;  363          0354  3     BEGIN
;  364          0355  3     setpro_mask = .setpro_mask OR %X'0F00';
;  365          0356  3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.GROUP',prot_desc)
;  366          0357  3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^8;
;  367          0358  2     END;
;  368          0359  2 IF cli$present(%ASCID 'OPTION.PROTECTION.WORLD')
;  369          0360  2 THEN
;  370          0361  3     BEGIN
;  371          0362  3     setpro_mask = .setpro_mask OR %X'F000';
;  372          0363  3     IF cli$get_value(%ASCID 'OPTION.PROTECTION.WORLD',prot_desc)
;  373          0364  3     THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^12;
;  374          0365  2     END;
;  375          0366  2
;  376          0367  2 !
;  377          0368  2 ! Complement the protection value since at this point, a bit set true
;  378          0369  2 ! indicates that we want to ALLOW access, while the system convention
;  379          0370  2 ! is that a bit set true indicates that we want to DENY access.
;  380          0371  2 !
;  381          0372  2 IF .setpro_mask NEQ 0                  ! If any protections specified
;  382          0373  2 THEN setpro_prot = NOT .setpro_prot;   ! then get the complement
;  383          0374  2
;  384          0375  2 !
;  385          0376  2 ! Now save the command level protection in the global protection
;  386          0377  2 ! area so it can be combined with individual file specifications
;  387          0378  2 ! later.  If the user did not supply a command level protection then
;  388          0379  2 ! the global_mask will have a value of zero.  If no local protection
```

```
;  389          0380  2 ! options are specified then the final result_mask will also have a
;  390          0381  2 ! value of zero.  This will cause the protection of the target file
;  391          0382  2 ! to be set to the current process default.
;  392          0383  2 !
;  393          0384  2
;  394          0385  2 global_mask = .setpro_mask;
;  395          0386  2 global_prot = .setpro_prot;
;  396          0387  2
;  397          0388  2 !
;  398          0389  2 !       Parse command qualifiers
;  399          0390  2 !
;  400          0391  2 qualifier$flags[qual_log] = cli$present(%ASCID 'LOG');
;  401          0392  2 qualifier$flags[qual_confirm] = cli$present(%ASCID 'CONFIRM');
;  402          0393  2
;  403          0394  2 !
;  404          0395  2 ! Loop thru all output files specified
;  405          0396  2 !
;  406          0397  2 WHILE get_output_file () DO              ! For each output file
;  407          0398  3     BEGIN
;  408          0399  3         lib$file_scan (                 ! Search for file names
;  409          0400  3                 output_fab,             !  -using this file name
;  410          0401  3                 setpro_action,          !  -call here on match
;  411          0402  3                 search_error)           !  -call here on error
;  412          0403  3
;  413          0404  2     END;
;  414          0405  2
;  415          0406  2 RETURN;
;  416          0407  2
;  417          0408  1 END;


                                                                .TITLE  SETPRO
                                                                .IDENT  \V04-000\

                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                    00  00  45  43  49  56  45  44  00000 P.AAB:   .ASCII  \DEVICE\<0><0>
                                                        010E0006  00008 P.AAA:   .LONG   17694726
                                                        00000000' 0000C         .ADDRESS P.AAB
49  54  43  45  54  4F  52  50  2E  4E  4F  49  54  50  4F  00010 P.AAD:   .ASCII  \OPTION.PROTECTION.SYSTEM\
                        4D  45  54  53  59  53  2E  4E  4F  0001F
                                                        010E0018  00028 P.AAC:   .LONG   17694744
                                                        00000000' 0002C         .ADDRESS P.AAD
49  54  43  45  54  4F  52  50  2E  4E  4F  49  54  50  4F  00030 P.AAF:   .ASCII  \OPTION.PROTECTION.SYSTEM\
                        4D  45  54  53  59  53  2E  4E  4F  0003F
                                                        010E0018  00048 P.AAE:   .LONG   17694744
                                                        00000000' 0004C         .ADDRESS P.AAF
49  54  43  45  54  4F  52  50  2E  4E  4F  49  54  50  4F  00050 P.AAH:   .ASCII  \OPTION.PROTECTION.OWNER\<0>
                        00  52  45  4E  57  4F  2E  4E  4F  0005F
                                                        010E0017  00068 P.AAG:   .LONG   17694743
                                                        00000000' 0006C         .ADDRESS P.AAH
49  54  43  45  54  4F  52  50  2E  4E  4F  49  54  50  4F  00070 P.AAJ:   .ASCII  \OPTION.PROTECTION.OWNER\<0>
                        00  52  45  4E  57  4F  2E  4E  4F  0007F
                                                        010E0017  00088 P.AAI:   .LONG   17694743
                                                        00000000' 0008C         .ADDRESS P.AAJ
49  54  43  45  54  4F  52  50  2E  4E  4F  49  54  50  4F  00090 P.AAL:   .ASCII  \OPTION.PROTECTION.GROUP\<0>
                        00  50  55  4F  52  47  2E  4E  4F  0009F
```

```
                                                 010E0017  000A8 P.AAK:  .LONG   17694743
                                                 00000000' 000AC         .ADDRESS P.AAL
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F     000B0 P.AAN:  .ASCII  \OPTION.PROTECTION.GROUP\<0>
                  00 50 55 4F 52 47 2E 4E 4F     000BF
                                                 010E0017  000C8 P.AAM:  .LONG   17694743
                                                 00000000' 000CC         .ADDRESS P.AAN
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F     000D0 P.AAP:  .ASCII  \OPTION.PROTECTION.WORLD\<0>
                  00 44 4C 52 4F 57 2E 4E 4F     000DF
                                                 010E0017  000E8 P.AAO:  .LONG   17694743
                                                 00000000' 000EC         .ADDRESS P.AAP
49 54 43 45 54 4F 52 50 2E 4E 4F 49 54 50 4F     000F0 P.AAR:  .ASCII  \OPTION.PROTECTION.WORLD\<0>
                  00 44 4C 52 4F 57 2E 4E 4F     000FF
                                                 010E0017  00108 P.AAQ:  .LONG   17694743
                                                 00000000' 0010C         .ADDRESS P.AAR
                                       00 47 4F 4C 00110 P.AAT:  .ASCII  \LOG\<0>
                                                 010E0003  00114 P.AAS:  .LONG   17694723
                                                 00000000' 00118         .ADDRESS P.AAT
                              00 4D 52 49 46 4E 4F 43 0011C P.AAV:  .ASCII  \CONFIRM\<0>
                                                 010E0007  00124 P.AAU:  .LONG   17694727
                                                 00000000' 00128         .ADDRESS P.AAV

                                                                         .PSECT  $OWN$,NOEXE,2

                                                 00000000  00000 SETPRO$FLAGS:
                                                                 .LONG   0
                                                 00000000  00004 QUALIFIER$FLAGS:
                                                                 .LONG   0
                                                     0000  00008 SETPRO_PROT:
                                                                 .WORD   0
                                                     0000  0000A SETPRO_MASK:
                                                                 .WORD   0
                                                           0000C GLOBAL_MASK:
                                                                 .BLKB   2
                                                           0000E GLOBAL_PROT:
                                                                 .BLKB   2
                                                           00010 RESULT_MASK:
                                                                 .BLKB   2
                                                           00012 RESULT_PROT:
                                                                 .BLKB   2
                                                           00014 OLDPRIV:.BLKB   8
                                                      00#  0001C NEWPRIV:.BYTE   0[3]
                                                      10   0001F          .BYTE   16
                                                           00020          .BLKB   4
                                                 00000001  00024 WORST_ERROR:
                                                                 .LONG   1
                                                           00028 OUTPUT_FILENAME:
                                                                 .BLKB   255
                                                           00127          .BLKB   1
                                                           00128 OUTPUT_NAM_RESULT:
                                                                 .BLKB   255
                                                           00227          .BLKB   1
                                                           00228 OUTPUT_NAM_EXPANDED:
                                                                 .BLKB   255
                                                           00327          .BLKB   1
                                                      02   00328 OUTPUT_RLF:
                                                                 .BYTE   2
                                                      60   00329          .BYTE   96
```

```
      00  0032A          .BYTE   0
      00  0032B          .BYTE   0
00000000  0032C          .LONG   0
      00  00330          .BYTE   0
      00  00331          .BYTE   0
      00  00332          .BYTE   0
      00  00333          .BYTE   0
00000000  00334          .LONG   0
00000000  00338          .LONG   0
    0000# 0033C          .WORD   0[8]
    0000# 0034C          .WORD   0[3]
    0000# 00352          .WORD   0[3]
00000000  00358          .LONG   0
00000000  0035C          .LONG   0
      00  00360          .BYTE   0
      00  00361          .BYTE   0
      00  00362          .BYTE   0
      00  00363          .BYTE   0
      00  00364          .BYTE   0
      00  00365          .BYTE   0
      00# 00366          .BYTE   0[2]
00000000  00368          .LONG   0
00000000  0036C          .LONG   0
00000000  00370          .LONG   0
00000000  00374          .LONG   0
00000000  00378          .LONG   0
00000000  0037C          .LONG   0
00000000# 00380          .LONG   0[2]
      02  00388 OUTPUT_NAM:
                         .BYTE   2
      60  00389          .BYTE   96
      FF  0038A          .BYTE   -1
      00  0038B          .BYTE   0
00000000' 0038C          .ADDRESS OUTPUT_NAM_RESULT
      00  00390          .BYTE   0
      00  00391          .BYTE   0
      FF  00392          .BYTE   -1
      00  00393          .BYTE   0
00000000' 00394          .ADDRESS OUTPUT_NAM_EXPANDED
00000000' 00398          .ADDRESS OUTPUT_RLF
    0000# 0039C          .WORD   0[8]
    0000# 003AC          .WORD   0[3]
    0000# 003B2          .WORD   0[3]
00000000  003B8          .LONG   0
00000000  003BC          .LONG   0
      00  003C0          .BYTE   0
      00  003C1          .BYTE   0
      00  003C2          .BYTE   0
      00  003C3          .BYTE   0
      00  003C4          .BYTE   0
      00  003C5          .BYTE   0
      00# 003C6          .BYTE   0[2]
00000000  003C8          .LONG   0
00000000  003CC          .LONG   0
00000000  003D0          .LONG   0
00000000  003D4          .LONG   0
00000000  003D8          .LONG   0
```

```
00000000  003DC          .LONG   0                                                       ;
00000000# 003E0          .LONG   0[2]                                                    ;
       03 003E8 OUTPUT_FAB:
                        .BYTE   3                                                       ;
       50 003E9         .BYTE   80                                                      ;
     0000 003EA         .WORD   0                                                       ;
00000000 003EC          .LONG   0                                                       ;
00000000 003F0          .LONG   0                                                       ;
00000000 003F4          .LONG   0                                                       ;
00000000 003F8          .LONG   0                                                       ;
     0000 003FC         .WORD   0                                                       ;
       02 003FE         .BYTE   2                                                       ;
       00 003FF         .BYTE   0                                                       ;
00000000 00400          .LONG   0                                                       ;
       00 00404         .BYTE   0                                                       ;
       00 00405         .BYTE   0                                                       ;
       00 00406         .BYTE   0                                                       ;
       02 00407         .BYTE   2                                                       ;
00000000 00408          .LONG   0                                                       ;
00000000 0040C          .LONG   0                                                       ;
00000000' 00410         .ADDRESS OUTPUT_NAM                                             ;
00000000 00414          .LONG   0                                                       ;
00000000 00418          .LONG   0                                                       ;
       00 0041C         .BYTE   0                                                       ;
       00 0041D         .BYTE   0                                                       ;
     0000 0041E         .WORD   0                                                       ;
00000000 00420          .LONG   0                                                       ;
     0000 00424         .WORD   0                                                       ;
       00 00426         .BYTE   0                                                       ;
       00 00427         .BYTE   0                                                       ;
00000000 00428          .LONG   0                                                       ;
00000000 0042C          .LONG   0                                                       ;
     0000 00430         .WORD   0                                                       ;
       00 00432         .BYTE   0                                                       ;
       00 00433         .BYTE   0                                                       ;
00000000 00434          .LONG   0                                                       ;

                        .EXTRN  CLI$GET_VALUE, CLI$PRESENT
                        .EXTRN  SYS$FAO, SYS$SETPRV
                        .EXTRN  LIB$GET_COMMAND
                        .EXTRN  LIB$FILE_SCAN, LIB$SET_FILE_PROT
                        .EXTRN  SET$DEVICE, SET$_PRONOTCHG

                        .PSECT  $CODE$,NOWRT,2

                 00FC 00000     .ENTRY  SET$PROTECTION, Save R2,R3,R4,R5,R6,R7          ; 0248
57      0000V    CF   9E 00002  MOVAB   PARSE_CLASS, R7                                 ;
56 00000000G     00   9E 00007  MOVAB   SYS$SETPRV, R6                                 ;
55 00000000G     00   9E 0000E  MOVAB   CLI$GET_VALUE, R5                              ;
54 00000000G     00   9E 00015  MOVAB   CLI$PRESENT, R4                                ;
53      0000'    CF   9E 0001C  MOVAB   SETPRO_PROT, R3                                ;
5E               08   C2 00021  SUBL2   #8, SP                                        ;
         0C      A3   9F 00024  PUSHAB  OLDPRIV                                       ; 0297
                 01   DD 00027  PUSHL   #1                                            ;
7E               01   7D 00029  MOVQ    #1, -(SP)                                    ;
66               04   FB 0002C  CALLS   #4, SYS$SETPRV                               ;
52               50   D0 0002F  MOVL    R0, STATUS                                   ;
```

```
                        15              52  E9 00032          BLBC    STATUS, 1$                                      ;
                1A      0F      A3      04  E0 00035          BBS     #4, OLDPRIV+3, 2$                               ; 0310
                                        7E  7C 0003A          CLRQ    -(SP)                                           ; 0313
                                    14  A3  9F 0003C          PUSHAB  NEWPRIV                                         ;
                                        7E  D4 0003F          CLRL    -(SP)                                           ;
                                    66      04  FB 00041      CALLS   #4, SYS$SETPRV                                  ;
                                    52      50  D0 00044      MOVL    R0, STATUS                                      ;
                                    0A      52  E8 00047      BLBS    STATUS, 2$                                      ;
                                            52  DD 0004A 1$:  PUSHL   STATUS                                        ; 0318
                            00000000G   00      01  FB 0004C      CALLS   #1, LIB$SIGNAL                              ;
                                                04  00053      RET                                                     ; 0317
                                    0000'   CF  9F 00054 2$:    PUSHAB  P.AAA                                       ; 0326
                                    64      01  FB 00058      CALLS   #1, CLI$PRESENT                                 ;
                                    0A      50  E9 0005B      BLBC    R0, 3$                                          ;
                                            01  DD 0005E      PUSHL   #1                                              ; 0329
                            00000000G   00      01  FB 00060      CALLS   #1, SET$DEVICE                              ;
                                                04  00067      RET                                                     ; 0328
                                    6E 020E0000     8F  D0 00068 3$:    MOVL    #34471936, PROT_DESC                ; 0336
                                            04      AE  D4 0006F      CLRL    PROT_DESC+4                             ;
                                    0000'   CF  9F 00072      PUSHAB  P.AAC                                       ; 0338
                                    64      01  FB 00076      CALLS   #1, CLI$PRESENT                                 ;
                                    18      50  E9 00079      BLBC    R0, 4$                                          ;
                                02  A3          0F  88 0007C      BISB2   #15, SETPRO_MASK                            ; 0341
                                            5E  DD 00080      PUSHL   SP                                              ; 0342
                                    0000'   CF  9F 00082      PUSHAB  P.AAE                                           ;
                                    65      02  FB 00086      CALLS   #2, CLI$GET_VALUE                               ;
                                    08      50  E9 00089      BLBC    R0, 4$                                          ;
                                            5E  DD 0008C      PUSHL   SP                                              ; 0343
                                    67      01  FB 0008E      CALLS   #1, PARSE_CLASS                                 ;
                                    63      50  B0 00091      MOVW    R0, SETPRO_PROT                                 ;
                                    0000'   CF  9F 00094 4$:    PUSHAB  P.AAG                                       ; 0345
                                    64      01  FB 00098      CALLS   #1, CLI$PRESENT                                 ;
                                    1C      50  E9 0009B      BLBC    R0, 5$                                          ;
                                02  A3          F0  8F  88 0009E      BISB2   #240, SETPRO_MASK                       ; 0348
                                            5E  DD 000A3      PUSHL   SP                                              ; 0349
                                    0000'   CF  9F 000A5      PUSHAB  P.AAI                                           ;
                                    65      02  FB 000A9      CALLS   #2, CLI$GET_VALUE                               ;
                                    0B      50  E9 000AC      BLBC    R0, 5$                                          ;
                                            5E  DD 000AF      PUSHL   SP                                              ; 0350
                                    67      01  FB 000B1      CALLS   #1, PARSE_CLASS                                 ;
                                    50      10  C4 000B4      MULL2   #16, R0                                         ;
                                    63      50  A8 000B7      BISW2   R0, SETPRO_PROT                                 ;
                                    0000'   CF  9F 000BA 5$:    PUSHAB  P.AAK                                       ; 0352
                                    64      01  FB 000BE      CALLS   #1, CLI$PRESENT                                 ;
                                    1C      50  E9 000C1      BLBC    R0, 6$                                          ;
                                03  A3          0F  88 000C4      BISB2   #15, SETPRO_MASK+1                          ; 0355
                                            5E  DD 000C8      PUSHL   SP                                              ; 0356
                                    0000'   CF  9F 000CA      PUSHAB  P.AAM                                           ;
                                    65      02  FB 000CE      CALLS   #2, CLI$GET_VALUE                               ;
                                    0C      50  E9 000D1      BLBC    R0, 6$                                          ;
                                            5E  DD 000D4      PUSHL   SP                                              ; 0357
                                    67      01  FB 000D6      CALLS   #1, PARSE_CLASS                                 ;
                50                  50      08  78 000D9      ASHL    #8, R0, R0                                      ;
                                    63      50  A8 000DD      BISW2   R0, SETPRO_PROT                                 ;
                                    0000'   CF  9F 000E0 6$:    PUSHAB  P.AAO                                       ; 0359
                                    64      01  FB 000E4      CALLS   #1, CLI$PRESENT                                 ;
                                    1D      50  E9 000E7      BLBC    R0, 7$                                          ;
```

```
                        03   A3        F0   8F   88 000EA          BISB2   #240, SETPRO_MASK+1                     ; 0362
                                       5E   DD 000EF              PUSHL   SP                                      ; 0363
                                 0000' CF   9F 000F1              PUSHAB  F.AAQ
                              65       02   FB 000F5              CALLS   #2, CLI$GET_VALUE
                              0C       50   E9 000F8              BLBC    R0, 7$
                                       5E   DD 000FB              PUSHL   SP                                      ; 0364
                              67       01   FB 000FD              CALLS   #1, PARSE_CLASS
                     50       50       0C   78 00100              ASHL    #12, R0, R0
                              63       50   A8 00104              BISW2   R0, SETPRO_PROT
                              50   02  A3   3C 00107 7$:          MOVZWL  SETPRO_MASK, R0                         ; 0372
                                       03   13 0010B              BEQL    8$
                              63       63   B2 0010D              MCOMW   SETPRO_PROT, SETPRO_PROT                ; 0373
                        04    A3       50   B0 00110 8$:          MOVW    R0, GLOBAL_MASK                         ; 0385
                        06    A3       63   B0 00114              MOVW    SETPRO_PROT, GLOBAL_PROT                ; 0386
                                 0000' CF   9F 00118              PUSHAB  P.AAS                                   ; 0391
                              64       01   FB 0011C              CALLS   #1, CLI$PRESENT
   FC   A3              01       01       50   F0 0011F              INSV    R0, #1, #1, QUALIFIER$FLAGS
                                 0000' CF   9F 00125              PUSHAB  P.AAU                                   ; 0392
                              64       01   FB 00129              CALLS   #1, CLI$PRESENT
   FC   A3              01       02       50   F0 0012C              INSV    R0, #2, #1, QUALIFIER$FLAGS
                     0000V    CF       00   FB 00132 9$:          CALLS   #0, GET_OUTPUT_FILE                     ; 0397
                              15       50   E9 00137              BLBC    R0, 10$
                                 0000V CF   9F 0013A              PUSHAB  SEARCH_ERROR                            ; 0399
                                 0000V CF   9F 0013E              PUSHAB  SETPRO_ACTION
                                 03E0  C3   9F 00142              PUSHAB  OUTPUT_FAB
                00000000G        00       03   FB 00146              CALLS   #3, LIB$FILE_SCAN
                                          E3   11 0014D              BRB     9$                                      ; 0398
                                               04 0014F 10$:        RET                                             ; 0408

; Routine Size:  336 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  419     0409  1 ROUTINE setpro_action (fab): =
;  420     0410  1
;  421     0411  1 !----
;  422     0412  1 !
;  423     0413  1 ! Functional description
;  424     0414  1 !
;  425     0415  1 !       This routine is called from lib$file_scan whenever
;  426     0416  1 !       a file name match occurs
;  427     0417  1 !
;  428     0418  1 ! Input parameters
;  429     0419  1 !
;  430     0420  1 !       fab = Address of block describing the file
;  431     0421  1 !       fab$l_nam = pointer to name block
;  432     0422  1 !
;  433     0423  1 ! Output parameters
;  434     0424  1 !
;  435     0425  1 !       First error encountered, or TRUE is RETURNed
;  436     0426  1 !
;  437     0427  1 !----
;  438     0428  1
;  439     0429  2 BEGIN
;  440     0430  2
;  441     0431  2 MAP fab: REF BBLOCK;                    ! Define fab block format
;  442     0432  2 BIND nam = .fab[fab$l_nam]: BBLOCK;     ! Define name block
;  443     0433  2
;  444     0434  2 LOCAL
;  445     0435  2     p_res_mask,                         ! Enable-mask parameter
;  446     0436  2     p_res_prot,                         ! Value-mask parameter
;  447     0437  2     final_prot: WORD,                   ! Recieves final protection
;  448     0438  2     desc:       VECTOR[2],              ! Temporary sting descriptor
;  449     0439  2     status;                             ! Recieves status
;  450     0440  2
;  451     0441  2 !
;  452     0442  2 ! If /CONFIRM was set by the user then interrogate him to see if
;  453     0443  2 ! this file is to be unlocked
;  454     0444  2 !
;  455     0445  2
;  456     0446  3 IF (.qualifier$flags[qual_confirm])     ! If confirmation requested
;  457     0447  2 THEN
;  458     0448  3         IF NOT (confirm_action (        ! Call confirmation rout. with
;  459     0449  3                 .fab))                  !  -address of fab
;  460     0450  2             THEN return(false);         ! If not confirmed then exit
;  461     0451  2
;  462     0452  2 !
;  463     0453  2 ! Now load the local descriptor with the file name size and address
;  464     0454  2 ! from the nam block
;  465     0455  2 !
;  466     0456  2
;  467     0457  2 desc[0] = .nam[nam$b_rsl];              ! Load file name size
;  468     0458  2 desc[1] = .nam[nam$l_rsa];              ! Load file name address
;  469     0459  2
;  470     0460  2 !
;  471     0461  2 ! But if this is a non-wildcarded network specification, then filescan
;  472     0462  2 ! doesn't do the $SEARCH, so we use the expanded file specification instead.
;  473     0463  2 !
;  474     0464  2
;  475     0465  2 IF .nam [nam$b_rsl] EQL 0
```

```
;  476          0466  3 THEN BEGIN
;  477          0467  3         desc[0] = .nam[nam$b_esl];      ! Load file name size
;  478          0468  3         desc[1] = .nam[nam$l_esa];      ! Load file name address
;  479          0469  2      END;
;  480          0470  2
;  481          0471  2
;  482          0472  2 !
;  483          0473  2 ! Compute the parameters for lib$set_file_prot.  If no /PROTECTION value was
;  484          0474  2 ! specified, call with null parameters for enable-mask and value-mask to cause
;  485          0475  2 ! protection to be set to the process default.
;  486          0476  2 !
;  487          0477  2
;  488          0478  2 p_res_mask = result_mask;                ! Assume not default
;  489          0479  2 p_res_prot = result_prot;
;  490          0480  2 IF .result_mask EQL 0                    ! If no protection values specified,
;  491          0481  2     THEN p_res_mask = p_res_prot = 0;    !     pass null parameters
;  492          0482  2
;  493          0483  2 !
;  494          0484  2 ! Call lib$set_file_prot to set file protection
;  495          0485  2 !
;  496          0486  2
;  497          0487  3 IF NOT (status = lib$set_file_prot (     ! Call library routine with
;  498          0488  3                              desc,       !  - file name
;  499          0489  3                              .p_res_mask,!  - result mask
;  500          0490  3                              .p_res_prot,!  - result protection
;  501          0491  3                              final_prot))!  - final protection returned
;  502          0492  3                                          !    by lib$set_file_prot
;  503          0493  2     THEN
;  504          0494  3         BEGIN
;  505        P 0495  3         write_message (                 ! Tell the user of error
;  506        P 0496  3                        set$_pronotchg,! - "Not changed" error message
;  507        P 0497  3                        1,             !  - 1 FAO arguement
;  508        P 0498  3                        desc,          !  - descriptor of filename
;  509          0499  3                        .status);      !  - original error
;  510          0500  3         return (.status);              ! Return to the caller
;  511          0501  2         END;
;  512          0502  2
;  513          0503  2
;  514          0504  2 !
;  515          0505  2 ! If /LOG was set then do it
;  516          0506  2 !
;  517          0507  2
;  518          0508  3 IF (.qualifier$flags[qual_log])          ! If logging requested
;  519          0509  2 THEN log_results (.fab,.final_prot);     !  then call the logger
;  520          0510  2
;  521          0511  2 RETURN (true);
;  522          0512  2
;  523          0513  1 END;


                                001C 00000 SETPRO_ACTION:
                                           .WORD   Save R2,R3,R4                          ; 0409
                5-      0000'   CF  9E 00002       MOVAB   QUALIFIER$FLAGS, R4            ;
                5E              0C  C2 00007       SUBL2   #12, SP                        ;
```

```
                    53      04   AC  D0 0000A          MOVL    FAB, R3                                 ; 0432
                    52      28   A3  D0 0000E          MOVL    40(R3), R2                              ;
      0A                    64       02  E1 00012      BBC     #2, QUALIFIER$FLAGS, 1$                 ; 0446
                                     53  DD 00016      PUSHL   R3                                      ; 0449
              0000V CF              01  FB 00018      CALLS   #1, CONFIRM_ACTION                      ;
                    68              50  E9 0001D      BLBC    R0, 6$                                  ;
                 04 AE      03      A2  9A 00020 1$:   MOVZBL  3(R2), DESC                             ; 0457
                 08 AE      04      A2  D0 00025       MOVL    4(R2), DESC+4                           ; 0458
                            03      A2  95 0002A       TSTB    3(R2)                                   ; 0465
                                    0A  12 0002D       BNEQ    2$                                      ;
                 04 AE      0B      A2  9A 0002F       MOVZBL  11(R2), DESC                            ; 0467
                 08 AE      0C      A2  D0 00034       MOVL    12(R2), DESC+4                          ; 0468
                    51      0C      A4  9E 00039 2$:   MOVAB   RESULT_MASK, P_RES_MASK                 ; 0478
                    50      0E      A4  9E 0003D       MOVAB   RESULT_PROT, P_RES_PROT                 ; 0479
                            0C      A4  B5 00041       TSTW    RESULT_MASK                             ; 0480
                                    02  12 00044       BNEQ    3$                                      ;
                                    50  7C 00046       CLRQ    P_RES_PROT                              ; 0481
                          4001      8F  BB 00048 3$:   PUSHR   #^M<R0,SP>                              ; 0490
                                    51  DD 0004C       PUSHL   P_RES_MASK                              ; 0489
                            10      AE  9F 0004E       PUSHAB  DESC                                    ; 0487
          00000000G         00          04  FB 00051      CALLS   #4, LIB$SET_FILE_PROT               ;
                    52              50  D0 00058       MOVL    R0, STATUS                              ;
                    18              52  E8 0005B       BLBS    STATUS, 4$                              ;
                                    52  DD 0005E       PUSHL   STATUS                                  ; 0499
                            08      AE  9F 00060       PUSHAB  DESC                                    ;
                                    01  DD 00063       PUSHL   #1                                      ;
                 00000000G          8F  DD 00065       PUSHL   #SET$_PRONOTCHG                         ;
          00000000G         00          04  FB 0006B      CALLS   #4, LIB$SIGNAL                      ;
                    50              52  D0 00072       MOVL    STATUS, R0                              ; 0500
                                        04 00075       RET                                             ;
      0A                    64       01  E1 00076 4$:   BBC     #1, QUALIFIER$FLAGS, 5$                 ; 0508
                    7E              6E  3C 0007A       MOVZWL  FINAL_PROT, -(SP)                       ; 0509
                                    53  DD 0007D       PUSHL   R3                                      ;
              0000V CF              02  FB 00080      CALLS   #2, LOG_RESULT                          ;
                    50              01  D0 00084 5$:   MOVL    #1, R0                                  ; 0511
                                        04 00087       RET                                             ;
                                    50  D4 00088 6$:   CLRL    R0                                      ; 0513
                                        04 0008A       RET                                             ;

; Routine Size:  139 bytes,    Routine Base:  $CODE$ + 0150
```

```
;  525          0514  1 ROUTINE get_output_file =
;  526          0515  1
;  527          0516  1 !----
;  528          0517  1 ! Functional Description
;  529          0518  1 !
;  530          0519  1 !       Obtain the next file specification from the output file
;  531          0520  1 !       list and perform the initial parsing to ensure legality.
;  532          0521  1 !
;  533          0522  1 ! Output Parameters
;  534          0523  1 !
;  535          0524  1 !       RETURNs TRUE if a file name was obtained
;  536          0525  1 !       RETURNS FALSE if the command line is exhausted
;  537          0526  1 !
;  538          0527  1 !       output_fab = FAB initialized for current file name
;  539          0528  1 !       output_nam = NAM block associated with FAB
;  540          0529  1 !
;  541          0530  1 !----
;  542          0531  1
;  543          0532  2 BEGIN
;  544          0533  2
;  545          0534  2 LOCAL
;  546          0535  2     prot_desc : $BBLOCK[dsc$c_s_bln],   ! Descriptor to define protection
;  547          0536  2     file_desc : $BBLOCK[dsc$c_s_bln];   ! Descriptor to define filename
;  548          0537  2
;  549          0538  2 CH$FILL(0, dsc$c_s_bln, file_desc);
;  550          0539  2 file_desc[dsc$w_length] = nam$c_maxrss;
;  551          0540  2 file_desc[dsc$a_pointer] = output_filename;
;  552          0541  2
;  553          0542  2 !
;  554          0543  2 !       If the output file list is depleted, then exit.
;  555          0544  2 !
;  556          0545  2 IF NOT cli$get_value(%ASCID 'FILE',file_desc)
;  557          0546  2 THEN RETURN false;
;  558          0547  2
;  559          0548  2 !
;  560          0549  2 ! Otherwise, continue processing.  Put new filename into FAB.
;  561          0550  2 !
;  562          0551  2 output_fab[fab$l_fna] = .file_desc[dsc$a_pointer];
;  563          0552  2 output_fab[fab$b_fns] = .file_desc[dsc$w_length];
;  564          0553  2
;  565          0554  2 !
;  566          0555  2 ! Get protection values (if any) for this file
;  567          0556  2 !
;  568          0557  2 setpro_prot = 0;          ! Clear any left over values
;  569          0558  2 setpro_mask = 0;          ! Clear any left over values
;  570          0559  2
;  571          0560  2 IF cli$present(%ASCID 'PROTECTION')
;  572          0561  2 THEN
;  573          0562  3     BEGIN
;  574          0563  3
;  575          0564  3     $init_dyndesc(prot_desc);   ! Make protection descriptor dynamic
;  576          0565  3
;  577          0566  3     IF cli$present(%ASCID 'PROTECTION.SYSTEM')
;  578          0567  3     THEN
;  579          0568  4         BEGIN
;  580          0569  4         setpro_mask = .setpro_mask OR %X'000F';
;  581          0570  4         IF cli$get_value(%ASCID 'PROTECTION.SYSTEM',prot_desc)
```

```
;  582          0571  4          THEN setpro_prot = parse_class(prot_desc);
;  583          0572  3          END;
;  584          0573  3      IF cli$present(%ASCID 'PROTECTION.OWNER')
;  585          0574  3      THEN
;  586          0575  4          BEGIN
;  587          0576  4          setpro_mask = .setpro_mask OR %X'00F0';
;  588          0577  4          IF cli$get_value(%ASCID 'PROTECTION.OWNER',prot_desc)
;  589          0578  4          THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^4;
;  590          0579  3          END;
;  591          0580  3      IF cli$present(%ASCID 'PROTECTION.GROUP')
;  592          0581  3      THEN
;  593          0582  4          BEGIN
;  594          0583  4          setpro_mask = .setpro_mask OR %X'0F00';
;  595          0584  4          IF cli$get_value(%ASCID 'PROTECTION.GROUP',prot_desc)
;  596          0585  4          THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^8;
;  597          0586  3          END;
;  598          0587  3      IF cli$present(%ASCID 'PROTECTION.WORLD')
;  599          0588  3      THEN
;  600          0589  4          BEGIN
;  601          0590  4          setpro_mask = .setpro_mask OR %X'F000';
;  602          0591  4          IF cli$get_value(%ASCID 'PROTECTION.WORLD',prot_desc)
;  603          0592  4          THEN setpro_prot = .setpro_prot OR parse_class(prot_desc)^12;
;  604          0593  3          END;
;  605          0594  3
;  606          0595  3 !
;  607          0596  3 ! Complement the protection value since at this point, a bit set true
;  608          0597  3 ! indicates that we want to ALLOW access, while the system convention
;  609          0598  3 ! is that a bit set true indicates that we want to DENY access.
;  610          0599  3 !
;  611          0600  3      setpro_prot = NOT .setpro_prot;
;  612          0601  2      END;
;  613          0602  2
;  614          0603  2 !
;  615          0604  2 ! Build the combined protection values.  That is, apply the command
;  616          0605  2 ! level protection first, and then any local protection values
;  617          0606  2 ! supplied with this parameter
;  618          0607  2 !
;  619          0608  2 result_mask =   .global_mask OR .setpro_mask;          ! OR masks
;  620          0609  3 result_prot =   (.global_prot AND (NOT .setpro_mask))  ! -global part
;  621          0610  2                          OR                           ! -OR'd with
;  622          0611  2                 (.setpro_prot AND .setpro_mask);      ! -local part
;  623          0612  2
;  624          0613  2 RETURN true;
;  625          0614  2
;  626          0615  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                  45  4C  49  46  0012C P.AAX:  .ASCII  \FILE\                                  ;
                                                                  010E0004  00130 P.AAW:  .LONG   17694724                        ;
                                                                  00000000' 00134         .ADDRESS P.AAX                          ;
                          00  00  4E  4F  49  54  43  45  54  4F  52  50  00138 P.AAZ:  .ASCII  \PROTECTION\<0><0>                 ;
                                                                  010E000A  00144 P.AAY:  .LONG   17694730                        ;
                                                                  00000000' 00148         .ADDRESS P.AAZ                          ;
54  53  59  53  2E  4E  4F  49  54  43  45  54  4F  52  50  0014C P.ABB:  .ASCII  \PROTECTION.SYSTEM\<0><0><0>                    ;
```

```
                                                00  00  00  4D  45  0015B
                                                          010E0011  00160 P.ABA:  .LONG   17694737
                                                          00000000' 00164         .ADDRESS P.ABB
54  53  59  53  2E  4E  4F  49  54  43  45  54  4F  52  50  00168 P.ABD:  .ASCII  \PROTECTION.SYSTEM\<0><0><0>
                                                00  00  00  4D  45  00177
                                                          010E0011  0017C P.ABC:  .LONG   17694737
                                                          00000000' 00180         .ADDRESS P.ABD
45  4E  57  4F  2E  4E  4F  49  54  43  45  54  4F  52  50  00184 P.ABF:  .ASCII  \PROTECTION.OWNER\
                                                                52  00193
                                                          010E0010  00194 P.ABE:  .LONG   17694736
                                                          00000000' 00198         .ADDRESS P.ABF
45  4E  57  4F  2E  4E  4F  49  54  43  45  54  4F  52  50  0019C P.ABH:  .ASCII  \PROTECTION.OWNER\
                                                                52  001AB
                                                          010E0010  001AC P.ABG:  .LONG   17694736
                                                          00000000' 001B0         .ADDRESS P.ABH
55  4F  52  47  2E  4E  4F  49  54  43  45  54  4F  52  50  001B4 P.ABJ:  .ASCII  \PROTECTION.GROUP\
                                                                50  001C3
                                                          010E0010  001C4 P.ABI:  .LONG   17694736
                                                          00000000' 001C8         .ADDRESS P.ABJ
55  4F  52  47  2E  4E  4F  49  54  43  45  54  4F  52  50  001CC P.ABL:  .ASCII  \PROTECTION.GROUP\
                                                                50  001DB
                                                          010E0010  001DC P.ABK:  .LONG   17694736
                                                          00000000' 001E0         .ADDRESS P.ABL
4C  52  4F  57  2E  4E  4F  49  54  43  45  54  4F  52  50  001E4 P.ABN:  .ASCII  \PROTECTION.WORLD\
                                                                44  001F3
                                                          010E0010  001F4 P.ABM:  .LONG   17694736
                                                          00000000' 001F8         .ADDRESS P.ABN
4C  52  4F  57  2E  4E  4F  49  54  43  45  54  4F  52  50  001FC P.ABP:  .ASCII  \PROTECTION.WORLD\
                                                                44  0020B
                                                          010E0010  0020C P.ABO:  .LONG   17694736
                                                          00000000' 00210         .ADDRESS P.ABP

                                                .PSECT  $CODE$,NOWRT,2

                                        07FC 00000 GET_OUTPUT_FILE:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 0514
                                5A      0000V   CF  9E 00002    MOVAB   PARSE_CLASS, R10        ;
                                59 00000000G    00  9E 00007    MOVAB   CLI$PRESENT, R9         ;
                                58 00000000G    00  9E 0000E    MOVAB   CLI$GET_VALUE, R8       ;
                                57      0000'   CF  9E 00015    MOVAB   P.AAW, R7               ;
                                56      0000'   CF  9E 0001A    MOVAB   SETPRO_PROT, R6         ;
                                5E              10  C2 0001F    SUBL2   #16, SP                 ;
08              00              6E              00  2C 00022    MOVC5   #0, (SP), #0, #8, FILE_DESC     ; 0538
                                                6E      00027   ;                               ;
                                6E        FF    8F  9B 00028    MOVZBW  #255, FILE_DESC         ; 0539
                          04    AE        20    A6  9E 0002C    MOVAB   OUTPUT_FILENAME, FILE_DESC+4    ; 0540
                                        4080    8F  BB 00031    PUSHR   #^M<R7,SP>              ; 0545
                                68              02  FB 00035    CALLS   #2, CLI$GET_VALUE       ;
                                03              50  E8 00038    BLBS    R0, 1$                  ;
                                            00E8    31 0003B    BRW     8$                      ;
                          040C  C6        04    AE  D0 0003E 1$:  MOVL    FILE_DESC+4, OUTPUT_FAB+44      ; 0551
                          0414  C6              6E  90 00044    MOVB    FILE_DESC, OUTPUT_FAB+52        ; 0552
                                                66  D4 00049    CLRL    SETPRO_PROT             ; 0557
                                        14      A7  9F 0004B    PUSHAB  P.AAY                   ; 0560
                                69              01  FB 0004E    CALLS   #1, CLI$PRESENT         ;
```

```
                03              50    E8 00051                BLBS    R0, 2$
                              00A7    31 00054                BRW     7$
            08  AE  020E0000     8F    D0 00057 2$:            MOVL    #34471936, PROT_DESC                    ;  0564
                          0C    AE    D4 0005F                CLRL    PROT_DESC+4
                          30    A7    9F 00062                PUSHAB  P.ABA                                   ;  0566
                69              01    FB 00065                CALLS   #1, CLI$PRESENT
                19              50    E9 00068                BLBC    R0, 3$
            02  A6              0F    88 0006B                BISB2   #15, SETPRO_MASK                        ;  0569
                          08    AE    9F 0006F                PUSHAB  PROT_DESC                               ;  0570
                          4C    A7    9F 00072                PUSHAB  P.ABC
                68              02    FB 00075                CALLS   #2, CLI$GET_VALUE
                09              50    E9 00078                BLBC    R0, 3$
                          08    AE    9F 0007B                PUSHAB  PROT_DESC                               ;  0571
                6A              01    FB 0007E                CALLS   #1, PARSE_CLASS
                66              50    B0 00081                MOVW    R0, SETPRO_PROT
                          64    A7    9F 00084 3$:            PUSHAB  P.ABE                                   ;  0573
                69              01    FB 00087                CALLS   #1, CLI$PRESENT
                1D              50    E9 0008A                BLBC    R0, 4$
            02  A6          F0   8F    88 0008D                BISB2   #240, SETPRO_MASK                       ;  0576
                          08    AE    9F 00092                PUSHAB  PROT_DESC                               ;  0577
                          7C    A7    9F 00095                PUSHAB  P.ABG
                68              02    FB 00098                CALLS   #2, CLI$GET_VALUE
                0C              50    E9 0009B                BLBC    R0, 4$
                          08    AE    9F 0009E                PUSHAB  PROT_DESC                               ;  0578
                6A              01    FB 000A1                CALLS   #1, PARSE_CLASS
                50              10    C4 000A4                MULL2   #16, R0
                66              50    A8 000A7                BISW2   R0, SETPRO_PROT
                        0094    C7    9F 000AA 4$:            PUSHAB  P.ABI                                   ;  0580
                69              01    FB 000AE                CALLS   #1, CLI$PRESENT
                1E              50    E9 000B1                BLBC    R0, 5$
            03  A6              0F    88 000B4                BISB2   #15, SETPRO_MASK+1                      ;  0583
                          08    AE    9F 000B8                PUSHAB  PROT_DESC                               ;  0584
                        00AC    C7    9F 000BB                PUSHAB  P.ABR
                68              02    FB 000BF                CALLS   #2, CLI$GET_VALUE
                0D              50    E9 000C2                BLBC    R0, 5$
                          08    AE    9F 000C5                PUSHAB  PROT_DESC                               ;  0585
                6A              01    FB 000C8                CALLS   #1, PARSE_CLASS
            50                  50    08 78 000CB             ASHL    #8, R0, R0
                66              50    A8 000CF                BISW2   R0, SETPRO_PROT
                        00C4    C7    9F 000D2 5$:            PUSHAB  P.ABM                                   ;  0587
                69              01    FB 000D6                CALLS   #1, CLI$PRESENT
                1F              50    E9 000D9                BLBC    R0, 6$
            03  A6          F0   8F    88 000DC                BISB2   #240, SETPRO_MASK+1                     ;  0590
                          08    AE    9F 000E1                PUSHAB  PROT_DESC                               ;  0591
                        00DC    C7    9F 000E4                PUSHAB  P.ABD
                68              02    FB 000E8                CALLS   #2, CLI$GET_VALUE
                0D              50    E9 000EB                BLBC    R0, 6$
                          08    AE    9F 000EE                PUSHAB  PROT_DESC                               ;  0592
                6A              01    FB 000F1                CALLS   #1, PARSE_CLASS
            50                  50    0C 78 000F4             ASHL    #12, R0, R0
                66              50    A8 000F8                BISW2   R0, SETPRO_PROT
                66              66    B2 000FB 6$:            MCOMW   SETPRO_PROT, SETPRO_PROT                ;  0600
08  A6          04  A6          02    A6 A9 000FE 7$:         BISW3   SETPRO_MASK, GLOBAL_MASK, RESULT_MASK   ;  0608
                51              06    A6 3C 00105             MOVZWL  GLOBAL_PROT, R1                         ;  0609
                50              02    A6 3C 00109             MOVZWL  SETPRO_MASK, R0
                51              50    CA 0010D                BICL2   R0, R1
                50              66    3C 00110                MOVZWL  SETPRO_PROT, R0                         ;  0611
```

```
                        52      02    A6  3C 00113          MOVZWL  SETPRO_MASK, R2                 ;
                        52            52  D2 00117          MCOML   R2, R2                          ;
                        50            52  CA 0011A          BICL2   R2, R0                          ;
                0A  A6  50            51  A9 0011D          BISW3   R1, R0, RESULT_PROT             ;
                        50            01  D0 00122          MOVL    #1, R0                          ; 0613
                                          04 00125          RET                                     ;
                                      50  D4 00126 8$:      CLRL    R0                              ; 0615
                                          04 00128          RET                                     ;

; Routine Size:  297 bytes,    Routine Base:  $CODE$ + 01DB
```

```
;  628     0616  1 ROUTINE confirm_action (fab) =
;  629     0617  1
;  630     0618  1 !----
;  631     0619  1 !
;  632     0620  1 ! Functional description
;  633     0621  1 !
;  634     0622  1 !       This routine is called from the main loop whenever
;  635     0623  1 !       confirmation is requested.
;  636     0624  1 !
;  637     0625  1 ! Input parameters
;  638     0626  1 !
;  639     0627  1 !       fab = Address of block describing the file
;  640     0628  1 !       fab$l_nam = pointer to name block
;  641     0629  1 !
;  642     0630  1 ! Output parameters
;  643     0631  1 !
;  644     0632  1 !               TRUE    --> Action should be taken
;  645     0633  1 !               FALSE   --> Action should be cancelled
;  646     0634  1 !
;  647     0635  1 !
;  648     0636  1 !----
;  649     0637  1
;  650     0638  2 BEGIN
;  651     0639  2
;  652     0640  2 MAP fab: REF BBLOCK;                    ! Define fab block format
;  653     0641  2 BIND nam = .fab[fab$l_nam]: BBLOCK;     ! Define name block
;  654     0642  2
;  655     0643  2 OWN
;  656     0644  2     file_desc:  BBLOCK[dsc$c_s_bln],    ! file_descriptor for file name
;  657     0645  2     fao_desc:   BBLOCK[dsc$c_s_bln],    ! FAO work area descriptor
;  658     0646  2     reply_desc: BBLOCK[dsc$c_s_bln]     ! Buffer desc. for reply
;  659     0647  2                 INITIAL (
;  660     0648  2                 WORD (0),               !  -size = 0
;  661     0649  2                 BYTE (dsc$k_dtype_t),   !  -arguement type = ascii
;  662     0650  2                 BYTE (dsc$k_class_d),   !  -class = dynamic
;  663     0651  2                 LONG (0));              !  -pointer = 0
;  664     0652  2 LOCAL
;  665     0653  2     status:     BLOCK[1],               ! Recieves status
;  666     0654  2     length :    WORD,                   ! Length of resultant message
;  667     0655  2     char,                               ! Character work area
;  668     0656  2     fao_buffer: VECTOR[512,BYTE];       ! FAO work area
;  669     0657  2
;  670     0658  2
;  671     0659  2 !
;  672     0660  2 ! Initialize descriptors with:
;  673     0661  2 !       1) FAO buffer   -->     fao_desc
;  674     0662  2 !       2) file name    -->     file_desc
;  675     0663  2 !
;  676     0664  2
;  677     0665  2 fao_desc[dsc$w_length] = 512;                   ! FAO work area size
;  678     0666  2 fao_desc[dsc$a_pointer] = fao_buffer;           ! FAO work area address
;  679     0667  2
;  680     0668  2 file_desc[dsc$w_length] = .nam[nam$b_rsl];      ! Resultant file name length
;  681     0669  2 file_desc[dsc$a_pointer] = .nam[nam$[_rsa];     ! Resultant file name address
;  682     0670  2
;  683     0671  2 IF .nam [nam$b_rsl] EQL 0                       ! If resultant name is blank
;  684     0672  3 THEN BEGIN                                     !    then use expanded name
```

```
;  685     0673  3      file_desc[dsc$w_length] = .nam[nam$b_esl]; !  - expanded file name length
;  686     0674  3      file_desc[dsc$a_pointer] = .nam[nam$l_esa];!  - expanded file name address
;  687     0675  2      END;
;  688     0676  2
;  689     0677  2
;  690     0678  2 !
;  691     0679  2 ! Now call SYS$FAO to expand message
;  692     0680  2 !
;  693     0681  2
;  694     0682  3 IF NOT (status = SYS$FAO (                ! Call system service with
;  695     0683  3                                           !  -message desc. address
;  696     0684  3                 addrdesc('!AS, change protection? (Y or N): '),
;  697     0685  3                 length,                   !  -place to put result length
;  698     0686  3                 fao_desc,                 !  -descriptor of a work area
;  699     0687  3                 file_desc))               !  -one FAO arguement
;  700     0688  2    THEN
;  701     0689  3         BEGIN
;  702     0690  3         write_message (.status);          ! Tell the user
;  703     0691  3         return (.status);                 ! Return to the caller
;  704     0692  2         END;
;  705     0693  2
;  706     0694  2 !
;  707     0695  2 ! Now question user by calling LIB$GET_COMMAND using the result
;  708     0696  2 ! of the FAO call as a prompt string
;  709     0697  2 !
;  710     0698  2
;  711     0699  2 fao_desc[dsc$w_length] = .length;         ! Move in prompt string size
;  712     0700  2
;  713     0701  3 IF NOT (status = LIB$GET_COMMAND (        ! Call library routine with
;  714     0702  3                 reply_desc,               !  -reply buffer desc. addr.
;  715     0703  3                 fao_desc))                !  -prompt string desc. addr.
;  716     0704  2    THEN
;  717     0705  3         BEGIN
;  718     0706  3         IF (.status EQL rms$_eof)         ! If this was end of file then
;  719     0707  3         THEN status[sts$v_severity] =     !  make it a severe (i.e. non-
;  720     0708  3                 sts$k_severe;             !  returnable) error
;  721     0709  3         write_message (.status);          ! Tell the user
;  722     0710  3         return (.status);                 ! Return to the caller
;  723     0711  2         END;
;  724     0712  2
;  725     0713  2
;  726     0714  2 !
;  727     0715  2 ! Now retrieve one charater from the buffer and set the return
;  728     0716  2 ! status depending on its value
;  729     0717  2 !
;  730     0718  2
;  731     0719  3 IF (.reply_desc[dsc$w_length] EQL 0)      ! Did user hit <CR>?
;  732     0720  2 THEN return (false);                      ! Yes, return false
;  733     0721  2
;  734     0722  2 char = CH$RCHAR (CH$PTR (.reply_desc[dsc$a_pointer]));
;  735     0723  2
;  736     0724  3 IF (.char EQL 'Y' OR                      ! If user responded 'Y'
;  737     0725  3     .char EQL 'y' OR                      !    or 'y'
;  738     0726  3     .char EQL '1' OR                      !    or '1' (TRUE)
;  739     0727  3     .char EQL 't')                        !    or 't'
;  740     0728  3 THEN return (true)                        ! Return 'TAKE ACTION'
;  741     0729  2 ELSE return (false);                      ! Return 'CANCEL ACTION'
```

```
;  742          0730  2
;  743          0731  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

6F 72 70 20 65 67 6E 61 68 63 20 2C 53 41 21  00214 P.ABR:  .ASCII  \.AS, change protection? (Y or N): \<0>
20 72 6F 20 59 28 20 3F 6E 6F 69 74 63 65 74  00223
                              00 20 3A 29 4E  00232
                                          00  00237        .ASCII  <0>
                                  00000022    00238 P.ABQ:  .LONG   34
                                  00000000'   0023C        .ADDRESS P.ABR

                                                        .PSECT  $OWN$,NOEXE,2

                                              00438 FILE_DESC:
                                                        .BLKB   8
                                              00440 FAO_DESC:
                                                        .BLKB   8
                                        0000  00448 REPLY_DESC:
                                                        .WORD   0
                                          0E  0044A        .BYTE   14
                                          02  0044B        .BYTE   2
                                  00000000    0044C        .LONG   0


                                                        .PSECT  $CODE$,NOWRT,2

                                       000C 00000 CONFIRM_ACTION:
                                                        .WORD   Save R2,R3                              ; 0616
                    53      0000'  CF  9E 00002         MOVAB   FAO_DESC, R3                            ;
                    5E      FDFC   CE  9E 00007         MOVAB   -516(SP), SP                            ;
                    50        04   AC  D0 0000C         MOVL    FAB, R0                                 ; 0641
                    50        28   A0  D0 00010         MOVL    40(R0), R0                              ;
                    63      0200   8F  B0 00014         MOVW    #512, FAO_DESC                          ; 0665
                 04 A3        04   AE  9E 00019         MOVAB   FAO_BUFFER, FAO_DESC+4                  ; 0666
                 F8 A3        03   A0  9B 0001E         MOVZBW  3(R0), FILE_DESC                        ; 0668
                 FC A3        04   A0  D0 00023         MOVL    4(R0), FILE_DESC+4                      ; 0669
                              03   A0  95 00028         TSTB    3(R0)                                   ; 0671
                              0A   12  0002B         BNEQ    1$                                      ;
                 F8 A3        0B   A0  9B 0002D         MOVZBW  11(R0), FILE_DESC                       ; 0673
                 FC A3        0C   A0  D0 00032         MOVL    12(R0), FILE_DESC+4                     ; 0674
                              F8   A3  9F 00037 1$:     PUSHAB  FILE_DESC                               ; 0682
                              53   DD  0003A         PUSHL   R3                                      ;
                              08   AE  9F 0003C         PUSHAB  LENGTH                                  ;
                           0000'   CF  9F 0003F         PUSHAB  P.ABQ                                   ; 0684
00000000G                 00        04  FB 00043         CALLS   #4, SYS$FAO                             ;
                          52        50  D0 0004A         MOVL    R0, STATUS                              ;
                          23        52  E9 0004D         BLBC    STATUS, 2$                              ; 0682
                          63        6E  B0 00050         MOVW    LENGTH, FAO_DESC                        ; 0699
                                    53  DD 00053         PUSHL   R3                                      ; 0701
                              08    A3  9F 00055         PUSHAB  REPLY_DESC                              ;
00000000G                 00        02  FB 00058         CALLS   #2, LIB$GET_COMMAND                     ;
                          52        50  D0 0005F         MOVL    R0, STATUS                              ;
                          1B        52  E8 00062         BLBS    STATUS, 3$                              ;
```

```
                    0001827A   8F                52  D1 00065          CMPL    STATUS, #98938                          ; 0706
                                                  05  12 0006C          BNEQ    2$
          52        03         00                04  F0 0006E          INSV    #4, #0, #3, STATUS                      ; 0707
                                                  52  DD 00073 2$:      PUSHL   STATUS                                  ; 0709
                    00000000G  00                01  FB 00075          CALLS   #1, LIB$SIGNAL
                               50                52  D0 0007C          MOVL    STATUS, R0                              ; 0710
                                                     04 0007F          RET
                                           08    A3  B5 00080 3$:      TSTW    REPLY_DESC                              ; 0719
                                                 2C  13 00083          BEQL    5$
                               50          0C    B3  9A 00085          MOVZBL  @REPLY_DESC+4, CHAR                     ; 0722
                    00000059   8F                50  D1 00089          CMPL    CHAR, #89                               ; 0724
                                                 1B  13 00090          BEQL    4$
                    00000079   8F                50  D1 00092          CMPL    CHAR, #121                              ; 0725
                                                 12  13 00099          BEQL    4$
                    00000054   8F                50  D1 0009B          CMPL    CHAR, #84                               ; 0726
                                                 09  13 000A2          BEQL    4$
                    00000074   8F                50  D1 000A4          CMPL    CHAR, #116                              ; 0727
                                                 04  12 000AB          BNEQ    5$
                               50                01  D0 000AD 4$:      MOVL    #1, R0                                  ; 0729
                                                     04 000B0          RET
                                                 50  D4 000B1 5$:      CLRL    R0                                      ; 0731
                                                     04 000B3          RET

; Routine Size:  180 bytes,    Routine Base:  $CODE$ + 0304
```

```
;  745     0732  1 ROUTINE log_results (fab,final_prot): =
;  746     0733  1
;  747     0734  1 !----
;  748     0735  1 !
;  749     0736  1 ! Functional description
;  750     0737  1 !
;  751     0738  1 !       This routine is called from the main loop whenever
;  752     0739  1 !       logging is requested
;  753     0740  1 !
;  754     0741  1 ! Input parameters
;  755     0742  1 !
;  756     0743  1 !       fab = Address of block describing the file
;  757     0744  1 !       fab$l_nam = pointer to name block
;  758     0745  1 !
;  759     0746  1 ! Output parameters
;  760     0747  1 !
;  761     0748  1 !       First error encountered, or TRUE is RETURNed
;  762     0749  1 !
;  763     0750  1 !----
;  764     0751  1
;  765     0752  2 BEGIN
;  766     0753  2
;  767     0754  2 LITERAL
;  768     0755  2     pbufsize = 32;                      ! Buffer for generating string
;  769     0756  2
;  770     0757  2 MAP fab: REF BBLOCK;                    ! Define fab block format
;  771     0758  2
;  772     0759  2 BIND nam = .fab[fab$l_nam]: BBLOCK;     ! Define name block
;  773     0760  2
;  774     0761  2 LOCAL
;  775     0762  2     status:,                            ! Recieves status
;  776     0763  2     pbuf:       VECTOR[pbufsize,BYTE],  ! Place for protection string
;  777     0764  2     pdesc:      VECTOR[2],              ! Temporary string descriptor
;  778     0765  2     desc:       VECTOR[2],              ! Temporary string descriptor
;  779     0766  2     prot_table: VECTOR[4];              ! Protection string table
;  780     0767  2
;  781     0768  2 IF .nam[nam$b_rsl] NEQ 0                ! IF result string nonblank,
;  782     0769  3 THEN BEGIN
;  783     0770  3     desc[0] = .nam[nam$b_rsl];          ! then display it
;  784     0771  3     desc[1] = .nam[nam$l_rsa];
;  785     0772  3     END
;  786     0773  2 ELSE IF .nam[nam$b_esl] NEQ 0           ! Or if expanded name nonblank
;  787     0774  3 THEN BEGIN
;  788     0775  3     desc[0] = .nam[nam$b_esl];          ! then display it
;  789     0776  3     desc[1] = .nam[nam$l_esa];
;  790     0777  3     END
;  791     0778  3 ELSE BEGIN
;  792     0779  3     desc[0] = .fab[fab$b_fns];          ! Otherwise, use original
;  793     0780  3     desc[1] = .fab[fab$l_fna];          ! name string in FAB
;  794     0781  2     END;
;  795     0782  2
;  796     0783  2 !
;  797     0784  2 ! Now build the resultant protection string from the value passed
;  798     0785  2 ! in the call.
;  799     0786  2 !
;  800     0787  2
;  801     0788  2 expand_prot(                            ! Call sub with
```

```
;  802          0784  2          prot_table,                     !  -place for the result
;  803          0790  2          .final_prot);                   !  -final protection value
;  804          0791  2
;  805          0792  2 pdesc[0] = pbufsize;                     ! Initialize descriptor size
;  806          0793  2 pdesc[1] = pbuf;                         ! Initialize descriptor address
;  807          0794  2
;  808          0795  3 IF NOT (status =
;  809        P 0796  3 $FAOL   (                                ! Call system service with
;  810        P 0797  3          CTRSTR = ADDRDESC ('S:!AS,O:!AS,G:!AS,W:!AS'),  ! -FAO string
;  811        P 0798  3          OUTLEN = pdesc[0],              !  -place for resultant length
;  812        P 0799  3          OUTBUF = pdesc,                 !  -output buffer descriptor
;  813          0800  4          PRMLST = prot_table)            !  -address of list of args
;  814          0801  3 ) THEN BEGIN
;  815          0802  3          write_message (.status);         ! Oops, tell the user
;  816          0803  3          return (.status);                ! And exit immediately
;  817          0804  2          END;
;  818          0805  2
;  819        P 0806  2 write_message (set$_protected,           ! Inform user with
;  820        P 0807  2                  2,                      !  -two FAO arguements
;  821        P 0808  2                  desc,                   !  -file name
;  822          0809  2                  pdesc);                 !  -new protection
;  823          0810  2
;  824          0811  2 RETURN (true);
;  825          0812  2
;  826          0813  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

21 3A 47 2C 53 41 21 3A 4F 2C 53 41 21 3A 53  00240 P.ABT:    .ASCII  \S:!AS,O:!AS,G:!AS,W:!AS\<0>                    ;
                  00 53 41 21 3A 57 2C 53 41  0024F                                                             ;
                                              00000017  00258 P.ABS:    .LONG   23                                ;
                                              00000000' 0025C           .ADDRESS P.ABT                            ;

                                                                .EXTRN  SYS$FAOL

                                                                .PSECT  $CODE$,NOWRT,2

                                                    000C 00000 LOG_RESULTS:
                                                                .WORD   Save R2,R3                                  ; 0732
                                          53 00000000G   00  9E 00002           MOVAB   LIB$SIGNAL, R3            ;
                                          5E         C0  AE  9E 00009           MOVAB   -64(SP), SP               ;
                                          51         04  AC  D0 0000D           MOVL    FAB, R1                   ; 0759
                                          50         28  A1  D0 00011           MOVL    40(R1), R0                ;
                                                     03  A0  95 00015           TSTB    3(R0)                     ; 0768
                                                     0C  13 00018              BEQL    1$                        ;
                                       10 AE         03  A0  9A 0001A           MOVZBL  3(R0), DESC               ; 0770
                                       14 AE         04  A0  D0 0001F           MOVL    4(R0), DESC+4             ; 0771
                                                     1B  11 00024              BRB     3$                        ; 0768
                                                     0B  A0  95 00026 1$:      TSTB    11(R0)                    ; 0773
                                                     0C  13 00029              BEQL    2$                        ;
                                       10 AE         0B  A0  9A 0002B           MOVZBL  11(R0), DESC              ; 0775
                                       14 AE         0C  A0  D0 00030           MOVL    12(R0), DESC+4            ; 0776
                                                     0A  11 00035              BRB     3$                        ; 0773
                                       10 AE         34  A1  9A 00037 2$:      MOVZBL  52(R1), DESC              ; 0779
                                       14 AE         2C  A1  D0 0003C           MOVL    44(R1), DESC+4            ; 0780
```

```
                          08   AC  DD 00041 3$:     PUSHL   FINAL_PROT                      ; 0790
                          04   AE  9F 00044         PUSHAB  PROT_TABLE                      ; 0788
                0000V     CF       02  FB 00047     CALLS   #2, EXPAND_PROT                 ;
                  18      AE       20  DO 0004C     MOVL    #32, PDESC                      ; 0792
                  1C      AE   20  AE  9E 00050     MOVAB   PBUF, PDESC+4                   ; 0793
                               5E  DD 00055         PUSHL   SP                              ; 0800
                               1C  AE 9F 00057      PUSHAB  PDESC                           ;
                               20  AE 9F 0005A      PUSHAB  PDESC                           ;
                           0000'   CF 9F 0005D      PUSHAB  P.ABS                           ;
              00000000G    00       04  FB 00061     CALLS   #4, SYS$FAOL                    ;
                           52       50  DO 00068     MOVL    R0, STATUS                      ;
                           09       52  E8 0006B     BLBS    STATUS, 4$                      ;
                                    52  DD 0006E     PUSHL   STATUS                          ; 0802
                           63       01  FB 00070     CALLS   #1, LIB$SIGNAL                  ;
                           50       52  DO 00073     MOVL    STATUS, R0                      ; 0803
                                    04  00076       RET                                     ;
                               18   AE  9F 00077 4$:  PUSHAB  PDESC                          ; 0809
                               14   AE  9F 0007A     PUSHAB  DESC                            ;
                                    02  DD 0007D     PUSHL   #2                              ;
                          007712AB  8F  DD 0007F     PUSHL   #7803563                        ;
                           63       04  FB 00085     CALLS   #4, LIB$SIGNAL                  ;
                           50       01  DO 00088     MOVL    #1, R0                          ; 0811
                                    04  0008B       RET                                     ; 0813

; Routine Size:  140 bytes,    Routine Base:  $CODE$ + 03B8
```

```
;  828          0814  1 ROUTINE expand_prot (table, protection): =
;  829          0815  1
;  830          0816  1 !----
;  831          0817  1 !
;  832          0818  1 ! Functional description
;  833          0819  1 !
;  834          0820  1 !       This routine fills a given VECTOR with the addresses of
;  835          0821  1 !       strings corresponding to a given protection word.
;  836          0822  1 !
;  837          0823  1 ! Input parameters
;  838          0824  1 !
;  839          0825  1 !       table = Address of the table to be filled in.
;  840          0826  1 !       protection = Protection word.
;  841          0827  1 !
;  842          0828  1 ! Output parameters
;  843          0829  1 !
;  844          0830  1 !       table has been filled in with the addresses of descriptors
;  845          0831  1 !       of strings describing each type of user (SYS,OWN,GRP,WORLD).
;  846          0832  1 !
;  847          0833  1 !----
;  848          0834  1
;  849          0835  2 BEGIN
;  850          0836  2
;  851          0837  2 BIND
;  852          0838  2     prot_table = .table: VECTOR[4];     ! Table of addresses
;  853          0839  2
;  854          0840  2 OWN
;  855          0841  2     prot_values: VECTOR[16] INITIAL(    ! Protection descriptions
;  856          0842  2         ADDRDESC('RWED'),
;  857          0843  2         ADDRDESC('WED'),
;  858          0844  2         ADDRDESC('RED'),
;  859          0845  2         ADDRDESC('ED'),
;  860          0846  2         ADDRDESC('RWD'),
;  861          0847  2         ADDRDESC('WD'),
;  862          0848  2         ADDRDESC('RD'),
;  863          0849  2         ADDRDESC('D'),
;  864          0850  2         ADDRDESC('RWE'),
;  865          0851  2         ADDRDESC('WE'),
;  866          0852  2         ADDRDESC('RE'),
;  867          0853  2         ADDRDESC('E'),
;  868          0854  2         ADDRDESC('RW'),
;  869          0855  2         ADDRDESC('W'),
;  870          0856  2         ADDRDESC('R'),
;  871          0857  2         ADDRDESC(''));
;  872          0858  2
;  873          0859  2 INCR index FROM 0 TO 3 DO
;  874          0860  2     prot_table[.index] = .prot_values [.protection<.index*4,4>];
;  875          0861  2
;  876          0862  2 RETURN (true);          ! Always return true
;  877          0863  2
;  878          0864  1 END;

                                                .PSECT  $PLIT$,NOWRT,NOEXE,2
                                44  45  57  52  00260 P.ABV:  .ASCII  \RWED\                                    ;
```

```
                00000004  00264 P.ABU:   .LONG   4
                00000000' 00268          .ADDRESS P.ABV
00 44 45 57     0026C P.ABX:   .ASCII  \WED\<0>
                00000003  00270 P.ABW:   .LONG   3
                00000000' 00274          .ADDRESS P.ABX
00 44 45 52     00278 P.ABZ:   .ASCII  \RED\<0>
                00000003  0027C P.ABY:   .LONG   3
                00000000' 00280          .ADDRESS P.ABZ
00 00 44 45     00284 P.ACB:   .ASCII  \ED\<0><0>
                00000002  00288 P.ACA:   .LONG   2
                00000000' 0028C          .ADDRESS P.ACB
00 44 57 52     00290 P.ACD:   .ASCII  \RWD\<0>
                00000003  00294 P.ACC:   .LONG   3
                00000000' 00298          .ADDRESS P.ACD
00 00 44 57     0029C P.ACF:   .ASCII  \WD\<0><0>
                00000002  002A0 P.ACE:   .LONG   2
                00000000' 002A4          .ADDRESS P.ACF
00 00 44 52     002A8 P.ACH:   .ASCII  \RD\<0><0>
                00000002  002AC P.ACG:   .LONG   2
                00000000' 002B0          .ADDRESS P.ACH
00 00 00 44     002B4 P.ACJ:   .ASCII  \D\<0><0><0>
                00000001  002B8 P.ACI:   .LONG   1
                00000000' 002BC          .ADDRESS P.ACJ
00 45 57 52     002C0 P.ACL:   .ASCII  \RWE\<0>
                00000003  002C4 P.ACK:   .LONG   3
                00000000' 002C8          .ADDRESS P.ACL
00 00 45 57     002CC P.ACN:   .ASCII  \WE\<0><0>
                00000002  002D0 P.ACM:   .LONG   2
                00000000' 002D4          .ADDRESS P.ACN
00 00 45 52     002D8 P.ACP:   .ASCII  \RE\<0><0>
                00000002  002DC P.ACO:   .LONG   2
                00000000' 002E0          .ADDRESS P.ACP
00 00 00 45     002E4 P.ACR:   .ASCII  \E\<0><0><0>
                00000001  002E8 P.ACQ:   .LONG   1
                00000000' 002EC          .ADDRESS P.ACR
00 00 57 52     002F0 P.ACT:   .ASCII  \RW\<0><0>
                00000002  002F4 P.ACS:   .LONG   2
                00000000' 002F8          .ADDRESS P.ACT
00 00 00 57     002FC P.ACV:   .ASCII  \W\<0><0><0>
                00000001  00300 P.ACU:   .LONG   1
                00000000' 00304          .ADDRESS P.ACV
00 00 00 52     00308 P.ACX:   .ASCII  \R\<0><0><0>
                00000001  0030C P.ACW:   .LONG   1
                00000000' 00310          .ADDRESS P.ACX
                          00314 P.ACZ:   .BLKB   0
                00000000  00314 P.ACY:   .LONG   0
                00000000' 00318          .ADDRESS P.ACZ

                                         .PSECT  $OWN$,NOEXE,2

00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00450 PROT_VALUES:
                                         .ADDRESS P.ABU, P.ABW, P.ABY, P.ACA, P.ACC, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00468          P.ACE, P.ACG, P.ACI, P.ACK, P.ACM, P.ACO, -
                00000000' 00000000' 00000000' 00000000' 00480          P.ACQ, P.ACS, P.ACU, P.ACW, P.ACY
```

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

                                                0004 00000 EXPAND_PROT:
                                                        .WORD   Save R2                                 ; 0814
                                             50  D4 00002          CLRL    INDEX                       ; 0859
                52              50              02  78 00004 1$:    ASHL    #2, INDEX, R2               ; 0860
        51      08  AC          04              52  EF 00008        EXTZV   R2, #4, PROTECTION, R1      ;
                        04 BC40     0000'CF41   D0 0000E            MOVL    PROT_VALUES[R1], @TABLE[INDEX] ;
                EA              50              03  F3 00016        AOBLEQ  #3, INDEX, 1$               ;
                                50              01  D0 0001A        MOVL    #1, R0                      ; 0862
                                                    04 0001D        RET                                 ; 0864

; Routine Size:  30 bytes,    Routine Base:  $CODE$ + 0444
```

```
;  880          0865  1 ROUTINE parse_class (desc) =
;  881          0866  1
;  882          0867  1 !---
;  883          0868  1 !
;  884          0869  1 ! This routine parses one class of user (e.g. SYSTEM, OWNER, GROUP, WORLD)
;  885          0870  1 ! to see what protection is allowed.  The value returned in the low 4 bits
;  886          0871  1 ! is the protection code, with the bits set to reflect that access is
;  887          0872  1 ! requested.  Note that this is exactly the opposite of what the system wants.
;  888          0873  1 !
;  889          0874  1 ! Inputs:
;  890          0875  1 !
;  891          0876  1 !       DESC -- a descriptor pointing to the ASCII representation of the
;  892          0877  1 !               protection desired
;  893          0878  1 !
;  894          0879  1 !---
;  895          0880  1
;  896          0881  2 BEGIN
;  897          0882  2
;  898          0883  2 MAP desc : REF $BBLOCK;
;  899          0884  2
;  900          0885  2 LOCAL
;  901          0886  2     pointer,                            ! Pointer to string
;  902          0887  2     result;                             ! Resultant protection
;  903          0888  2
;  904          0889  2 !
;  905          0890  2 ! Initially set the value to all zeros, no access
;  906          0891  2 !
;  907          0892  2 result = 0;
;  908          0893  2
;  909          0894  2 !
;  910          0895  2 ! Scan for the occurrence of each keyletter, and, if it is there, set the
;  911          0896  2 ! appropriate bit.
;  912          0897  2 !
;  913          0898  2 pointer = .desc[dsc$a_pointer];
;  914          0899  2 INCR index FROM 1 to .desc[dsc$w_length] DO
;  915          0900  3     BEGIN
;  916          0901  3     LOCAL char : BYTE;
;  917          0902  3     char = CH$RCHAR_A(pointer);
;  918          0903  3     IF .char EQL'R'
;  919          0904  3     THEN result = .result OR %X'1'
;  920          0905  3     ELSE IF .char EQL  'W'
;  921          0906  3     THEN result = .result OR %X'2'
;  922          0907  3     ELSE IF .char EQL 'E'
;  923          0908  3          OR .char EQL 'P'
;  924          0909  3     THEN result = .result OR %X'4'
;  925          0910  3     ELSE IF .char EQL 'D'
;  926          0911  3          OR .char EQL 'L'
;  927          0912  3     THEN result = .result OR %X'8'
;  928          0913  3     ELSE SIGNAL_STOP (set$_syntax, 1, .desc);
;  929          0914  2     END;
;  930          0915  2
;  931          0916  2 RETURN .result;
;  932          0917  1 END;
```

```
                                007C 00000 PARSE_CLASS:
                                           .WORD   Save R2,R3,R4,R5,R6                     ; 0865
                          52    04   AC  D0 00002          MOVL    DESC, R2                ; 0898
                          56    04   A2  D0 00006          MOVL    4(R2), POINTER
                          55         62  3C 0000A          MOVZWL  (R2), R5                ; 0899
                                     53  7C 0000D          CLRQ    INDEX
                                     4C  11 0000F          BRB     8$
                          50         86  90 00011 1$:      MOVB    (POINTER)+, CHAR        ; 0902
                     52   8F         50  91 00014          CMPB    CHAR, #82               ; 0903
                                     05  12 00018          BNEQ    2$
                          54         01  88 0001A          BISB2   #1, RESULT              ; 0904
                                     3E  11 0001D          BRB     8$
                     57   8F         50  91 0001F 2$:      CMPB    CHAR, #87               ; 0905
                                     05  12 00023          BNEQ    3$
                          54         02  88 00025          BISB2   #2, RESULT              ; 0906
                                     33  11 00028          BRB     8$
                     45   8F         50  91 0002A 3$:      CMPB    CHAR, #69               ; 0907
                                     06  13 0002E          BEQL    4$
                     50   8F         50  91 00030          CMPB    CHAR, #80               ; 0908
                                     05  12 00034          BNEQ    5$
                          54         04  88 00036 4$:      BISB2   #4, RESULT              ; 0909
                                     22  11 00039          BRB     8$
                     44   8F         50  91 0003B 5$:      CMPB    CHAR, #68               ; 0910
                                     06  13 0003F          BEQL    6$
                     4C   8F         50  91 00041          CMPB    CHAR, #76               ; 0911
                                     05  12 00045          BNEQ    7$
                          54         08  88 00047 6$:      BISB2   #8, RESULT              ; 0912
                                     11  11 0004A          BRB     8$
                                     52  DD 0004C 7$:      PUSHL   R2                      ; 0913
                                     01  DD 0004E          PUSH    #1
                               007710FC  8F  DD 00050          PUSHL   #7803132
                    00000000G    00         03  FB 00056          CALLS   #3, LIB$STOP
                          B0         53         55  F3 0005D 8$:      AOBLEQ  R5, INDEX, 1$   ; 0899
                                     50         54  D0 00061          MOVL    RESULT, R0      ; 0916
                                                    04 00064          RET                     ; 0917

; Routine Size:  101 bytes,    Routine Base:  $CODE$ + 0462
```

```
;  934          0918  1 ROUTINE search_error (fab): =
;  935          0919  1
;  936          0920  1 !----
;  937          0921  1 !
;  938          0922  1 ! Functional description
;  939          0923  1 !
;  940          0924  1 !       This routine is called from RMS whenever an error
;  941          0925  1 !       occurs during an RMS file function call.
;  942          0926  1 !
;  943          0927  1 ! Input parameters
;  944          0928  1 !
;  945          0929  1 !       fab = Address of block used in the RMS call.
;  946          0930  1 !       fab$l_nam = pointer to name block
;  947          0931  1 !
;  948          0932  1 ! Output parameters
;  949          0933  1 !
;  950          0934  1 !       FAB status is returned
;  951          0935  1 !
;  952          0936  1 !----
;  953          0937  1
;  954          0938  2 BEGIN
;  955          0939  2
;  956          0940  2 MAP fab: REF BBLOCK;                    ! Define fab block format
;  957          0941  2 BIND nam = .fab[fab$l_nam]: BBLOCK;     ! Define name block
;  958          0942  2
;  959          0943  2 LOCAL desc: VECTOR[2];                  ! Temporary string descriptor
;  960          0944  2
;  961          0945  2 IF .nam[nam$b_rsl] NEQ 0                ! If result string nonblank,
;  962          0946  3 THEN BEGIN
;  963          0947  3     desc[0] = .nam[nam$b_rsl];          ! then display it
;  964          0948  3     desc[1] = .nam[nam$l_rsa];
;  965          0949  3     END
;  966          0950  2 ELSE IF .nam[nam$b_esl] NEQ 0           ! Or if expanded name nonblank
;  967          0951  3 THEN BEGIN
;  968          0952  3     desc[0] = .nam[nam$b_esl];          ! then display it
;  969          0953  3     desc[1] = .nam[nam$l_esa];
;  970          0954  3     END
;  971          0955  3 ELSE BEGIN
;  972          0956  3     desc[0] = .fab[fab$b_fns];          ! Otherwise, use original
;  973          0957  3     desc[1] = .fab[fab$l_fna];          ! name string in FAB
;  974          0958  2     END;
;  975          0959  2
;  976        P 0960  2 write_message(set$_searchfail,1,DESC,   ! Output an error message
;  977        P 0961  2                .fab[fab$l_sts],        ! with fab error code
;  978          0962  2                .fab[fab$l_stv]);       ! and secondary code
;  979          0963  2
;  980          0964  2 RETURN (.fab[fab$l_sts]);
;  981          0965  2
;  982          0966  1 END;


                                0004 00000 SEARCH_ERROR:
                                           .WORD   Save R2                          ; 0918
                     5E         08   C2 00002       SUBL2   #8, SP                 ;
```

```
                          52        04    AC  D0 00005          MOVL    FAB, R2                          ; 0941
                          50        28    A2  D0 00009          MOVL    40(R2), R0                       ;
                                    03    A0  95 0000D          TSTB    3(R0)                            ; 0945
                                    0B    13  00010             BEQL    1$                               ;
                          6E        03    A0  9A 00012          MOVZBL  3(R0), DESC                      ; 0947
                    04    AE        04    A0  D0 00016          MOVL    4(R0), DESC+4                    ; 0948
                                    19    11  0001B             BRB     3$                               ; 0945
                                    0B    A0  95 0001D 1$:      TSTB    11(R0)                           ; 0950
                                    0B    13  00020             BEQL    2$                               ;
                          6E        0B    A0  9A 00022          MOVZBL  11(R0), DESC                     ; 0952
                    04    AE        0C    A0  D0 00026          MOVL    12(R0), DESC+4                   ; 0953
                                    09    11  0002B             BRB     3$                               ; 0950
                          6E        34    A2  9A 0002D 2$:      MOVZBL  52(R2), DESC                     ; 0956
                    04    AE        2C    A2  D0 00031          MOVL    44(R2), DESC+4                   ; 0957
                          7E        08    A2  7D 00036 3$:      MOVQ    8(R2), -(SP)                     ; 0962
                                    08    AE  9F 0003A          PUSHAB  DESC                             ;
                                          01  DD 0003D          PUSHL   #1                               ;
                              0077123A    8F  DD 0003F          PUSHL   #7803450                         ;
          00000000G           00          05  FB 00045          CALLS   #5, LIB$SIGNAL                   ;
                          50        08    A2  D0 0004C          MOVL    8(R2), R0                        ; 0964
                                          04  00050             RET                                      ; 0966

; Routine Size:  81 bytes,    Routine Base:  $CODE$ + 04C7
```

```
;  984          0967  1 ROUTINE cli_error (cli_block,error): =
;  985          0968  1
;  986          0969  1 !----
;  987          0970  1 !
;  988          0971  1 ! Functional description
;  989          0972  1 !
;  990          0973  1 !       This routine is called as an error handler for
;  991          0974  1 !       CLI errors.
;  992          0975  1 !
;  993          0976  1 ! Input parameters
;  994          0977  1 !
;  995          0978  1 !       cli_block = Address of CLI request block
;  996          0979  1 !       error = CLI error message
;  997          0980  1 !
;  998          0981  1 ! Output parameters
;  999          0982  1 !
; 1000          0983  1 !       Input 'error' is RETURNed
; 1001          0984  1 !
; 1002          0985  1 !----
; 1003          0986  1
; 1004          0987  2 BEGIN
; 1005          0988  2
; 1006          0989  2 MAP cli_block: REF BBLOCK;              ! Define CLI request block
; 1007          0990  2
; 1008          0991  2 signal_stop(.error);
; 1009          0992  2
; 1010          0993  2 RETURN (.error);
; 1011          0994  2
; 1012          0995  1 END;

                                0000 00000 CLI_ERROR:
                                           .WORD   Save nothing                    ; 0967
                           08   AC  DD 00002       PUSHL   ERROR                   ; 0991
                00000000G  00       01  FB 00005  CALLS   #1, LIB$STOP            ;
                           50   08   AC  D0 0000C  MOVL    ERROR, R0              ; 0993
                                     04 00010      RET                            ; 0995

; Routine Size:  17 bytes,    Routine Base:  $CODE$ + 0518
```

```
; 1014          0996  1 END
; 1015          0997  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  $OWN$                        1168  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                        796  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       1321  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                           Library Statistics
;
;                                    -------- Symbols --------      Pages       Processing
;       File                         Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1   9776        58         0       581          00:01.0


;                                       COMMAND QUALIFIERS

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

; Size:         1321 code + 1964 data bytes
; Run Time:       00:27.2
; Elapsed Time:   01:59.6
; Lines/CPU Min:   2202
; Lexemes/CPU-Min: 21945
; Memory Used:  154 pages
; Compilation Complete
```

SETFILE
LIS

SETP0MESS
LIS

SETP0DISP
LIS

SETMISC
LIS

SETPRO
LIS

SETMAIN
LIS

SETPROCES
LIS

SETSHOBRO
LIS

SETVOLUME
LIS

SETPWD
LIS

SETTERM
LIS

SETQUEUE
LIS

SETTIME
LIS