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

```
**FILE**ID**SHOWPROC                         F  9
```

```
  SSSSSSSS  HH      HH    OOOOOO    WW      WW  PPPPPPPP    RRRRRRRR      OOOOOO      CCCCCCCC
  SSSSSSSS  HH      HH    OOOOOO    WW      WW  PPPPPPPP    RRRRRRRR      OOOOOO      CCCCCCCC
SS          HH      HH  OO      OO  WW      WW  PP      PP  RR      RR  OO      OO  CC
SS          HH      HH  OO      OO  WW      WW  PP      PP  RR      RR  OO      OO  CC
SS          HH      HH  OO      OO  WW      WW  PP      PP  RR      RR  OO      OO  CC
SS          HH      HH  OO      OO  WW      WW  PP      PP  RR      RR  OO      OO  CC
  SSSSSS    HHHHHHHHHH  OO      OO  WW      WW  PPPPPPPP    RRRRRRRR    OO      OO  CC
  SSSSSS    HHHHHHHHHH  OO      OO  WW      WW  PPPPPPPP    RRRRRRRR    OO      OO  CC
        SS  HH      HH  OO      OO  WW  WW  WW  PP          RR  RR      OO      OO  CC
        SS  HH      HH  OO      OO  WW  WW  WW  PP          RR  RR      OO      OO  CC
        SS  HH      HH  OO      OO  WWWW  WWWW  PP          RR    RR    OO      OO  CC              ....
        SS  HH      HH  OO      OO  WWWW  WWWW  PP          RR    RR    OO      OO  CC              ....
SSSSSSSS    HH      HH    OOOOOO    WW      WW  PP          RR      RR    OOOOOO      CCCCCCCC      ....
SSSSSSSS    HH      HH    OOOOOO    WW      WW  PP          RR      RR    OOOOOO      CCCCCCCC      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 MODULE showprocess (IDENT = 'V04-000',
0                     ADDRESSING_MODE (EXTERNAL = GENERAL)) =
0
1 BEGIN
1 !
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
1 !*  ALL RIGHTS RESERVED.                                                     *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
1 !*  TRANSFERRED.                                                             *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
1 !*  CORPORATION.                                                             *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1 !
1 !++
1 !
1 ! FACILITY:  SHOW utility
1 !
1 ! ABSTRACT:
1 !       This module contains the routines for the SHOW PROCESS command.
1 !
1 ! ENVIRONMENT:
1 !       VAX native, user and kernel mode
1 !
1 ! AUTHOR:  Gerry Smith           CREATION DATE:   8-Sep-1982
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-022 MCN0188         Maria del C. Nasr       29-Aug-1984
1 !               Add routine CHECK_PRIVELEGE to turn off WORLD privelege
1 !               if the user issuing the SHOW PROCESS command does not
1 !               have it.
1 !
1 !       V03-021 MCN0182         Maria del C. Nasr       24-Jul-1984
1 !               Add three new quotas returned by GETJPI: MAXDETACH,
1 !               MAXJOBS, SHRFILLM.  Also, make the sizes in the jpi
1 !               list match those in $JPIDEF macro.
1 !
1 !       V03-020 MCN0181         Maria del C. Nasr       23-Jul-1984
1 !               Eliminate from the display of SHOW PROC/PRIV some
1 !               privileges that are not implemented yet (UPGRADE,
!               DOWNGRADE, PRMJNL, TMPJNL).
!
!       V03-019 MCN0178         Maria del C. Nasr       16-Jul-1984
!               Fix misspellings in messages displayed by SHOW PROC/PRIV.
!
!       V03-018 AEW0001         Anne E. Warner          17-May-1984
!               Increase the size of the variable that holds the
!               logical translation of the process default disk
!               (SYS$DISK) to 256 bytes.  This is to avoid truncation
!               of the disk name when printed.
!
!       V03-017 HH0002          Hai Huang               27-Feb-1984
!               Add job-wide mount support.
!
!       V03-016 MCN0149         Maria del C. Nasr       08-Feb-1984
!               If SHOW PROCESS/ALL/ID is specified and the process id
!               is not the same as the current process, do not display
!               memory, nor subprocesses.  Also, when the user specified
!               a pid value, call $GETJPI to make sure we work with the
!               extended form.
!
!       V03-015 GAS0187         Gerry Smith             20-Sep-1983
!               Fix up displays for /quota and /accounting.
!
!       V03-014 LMP0140         L. Mark Pilant,         23-Aug-1983  22:43
!               Add support for alphanumeric UICs.
!
!       V03-013 GAS0159         Gerry Smith             26-Jul-1983
!               Fix the display for SHOW PROC/PRIV so that the
!               output is columnar.
!
!       V03-012 GAS0149         Gerry Smith             27-Jun-1983
!               Obtain device names from the common system routine
!               IOC$CVT_DEVNAM.
!
!       V03-011 RSH0020         R. Scott Hanna          21-May-1983
!               Add SECURITY privilege
!
!       V03-010 GAS0131         Gerry Smith             17-May-1983
!               Add the access control rights lists for SHOW PROC/PRIV.
!
!       V03-009 GAS0129         Gerry Smith             28-Apr-1983
!               For MSCP devices, fix display problem that caused the
!               unit number to be zero always.
!
!       V03-008 GAS0124         Gerry Smith             20-Apr-1983
!               Change the name of prv$v_noacnt to ACNT.  Also add
!               TMPJNL and PRMJNL.
!
!       V03-07  GAS0115         Gerry Smith              4-Apr-1983
!               Add support for cluster devices.
!
!       V03-006 LMP0083         L. Mark Pilant,         28-Feb-1983  15:27
!               Add support for the new privileges: SHARE, UPGRADE, DOWNGRADE,
!               GRPPRV, and READALL.
!
!       V03-005 CWH1002         CW Hobbs                25-Feb-1983
```

```
1 !                 Use a routine to convert the extended pid to a pcb address.
1 !                 Also convert extended pid to ipid to check device allocation.
1 !                 Use SCH$GL_PIXWIDTH to reference pix in epid.  Fix a couple of
1 !                 4 byte iosb's to be 8 bytes.  Use literal SCH$C_SWPPIX to anchor
1 !                 loop through PCBVEC.
1 !
1 !         V03-004 GAS0107         Gerry Smith             24-Jan-1983
1 !                 If an error occurs with the /ID qualifier, signal
1 !                 a more reasonable message.
1 !
1 !         V03-003 GAS0106         Gerry Smith             21-Jan-1983
1 !                 Change the name SETPRI to ALTPRI, since this is the
1 !                 preferred name.
1 !
1 !         V03-002 GAS0104         Gerry Smith             21-Jan-1983
1 !                 Fix the privilege display so that the scan doesn't
1 !                 go beyond the end of the privilege table.
1 !
1 !         V03-001 GAS0098         Gerry Smith             7-Jan-1983
1 !                 Add the count of images activated.
1 !
1 !--
```

```

!
! Include files
!

LIBRARY 'SYS$LIBRARY:LIB';                   ! VAX/VMS system definitions
REQUIRE 'SRC$:SHOWDEF';                      ! SHOW common definitions

!
! Define shared messages.
!
$SHR_MSGDEF     (SHOW,120,LOCAL,
                (INVQUAVAL,ERROR));

!
! Define the linkage for the routines to lock, unlock, and scan  the I/O
! database, as well as the routines to manipulate pids.
!
LINKAGE
    IOLOCK = JSB (REGISTER = 4),
    CVTPID = JSB (REGISTER=0) : PRESERVE (1,2,3,4,5) NOTUSED (6,7,8,9,10,11),
    IOSCAN = JSB (REGISTER = 11,                   ! Call with DDB,
                  REGISTER = 10;                   ! UCB,
                  REGISTER = 11,                   ! Return with DDB,
                  REGISTER = 10),                  ! UCB
    CVTDEV = JSB (REGISTER = 0,                    ! Length of output buffer,
                  REGISTER = 1,                    ! Address of output buffer
                  REGISTER = 4,                    ! Format of device name
                  REGISTER = 5;                    ! Address of UCB
                  REGISTER = 1);                   ! Length of final name



!
! Define bits for the flags longword
!
MACRO
    PROC$V_DEF          =       0, 0, 1, 0%,    ! print default info
    PROC$V_QUOT         =       0, 1, 1, 0%,    ! /QUOTAS
    PROC$V_ACC          =       0, 2, 1, 0%,    ! /ACCOUNTING
    PROC$V_PRIV         =       0, 3, 1, 0%,    ! /PRIVILEGES
    PROC$V_MEM          =       0, 4, 1, 0%,    ! /MEMORY
    PROC$V_SUB          =       0, 5, 1, 0%,    ! /SUBPROCESS
    PROC$V_ALL          =       0, 6, 1, 0%,    ! /ALL
    PROC$V_ID           =       0, 7, 1, 0%,    ! /IDENTIFICATION
    PROC$V_CONT         =       0, 8, 1, 0%;    ! /CONTINUOUS

!
! Define the locations of the device name and unit number for allocated
! and mou ted devices.
!
MACRO
    d_l_length  = 0, 0, 32, 0%,                    ! Length of device name
    d_a_ptr     = 4, 0, 32, 0%,                    ! Address of device name
    d_t_device  = 8, 0,  8, 0%;                    ! ASCII device name
LITERAL d_k_length = $BYTEOFFSET(d_t_device) + 21;

```

```
;  195          0292   1 !
;  196          0293   1 ! Define the locations of the owner pix and process pix in the
;  197          0294   1 ! structure that contains information about subprocesses.
;  198          0295   1 !
;  199          0296   1 MACRO
;  200          0297   1     sub_owner = 0, 0, 16, 0%,
;  201          0298   1     sub_pix   = 2, 0, 16, 0%,
;  202          0299   1     sub_name  = 4, 0,  8, 0%;
;  203          0300   1
;  204          0301   1 !
;  205          0302   1 ! A couple of macros to build the privilege table.
;  206          0303   1 !
;  207          0304   1 MACRO
;  208          0305   1     text_entry [text] = UPLIT(%ASCIC %STRING(text))%,
;  209          0306   1
;  210          0307   1
;  211        M 0308   1     make_priv_table (table_name) =
;  212        M 0309   1         LITERAL priv_num = (%LENGTH -1)/2;
;  213        M 0310   1         OWN priv_table : VECTOR[%LENGTH]
;  214          0311   1                         INITIAL ( text_entry(%REMAINING) );%;
;  215          0312   1
;  216          0313   1 !
;  217          0314   1 ! The following set of macros are used to get all the information about
;  218          0315   1 ! a particular process.  These macros produce:
;  219          0316   1 !       1. A list of arguments for $FAOL, which are used to output
;  220          0317   1 !          information.  The elements in this list are all prefixed
;  221          0318   1 !          with 'FAO_'
;  222          0319   1 !       2. A list of auxiliary data buffers.  The elements are all
;  223          0320   1 !          prefixed with 'AUX_'
;  224          0321   1 !       3. A GETJPI list which will store all the information requested,
;  225          0322   1 !          into the FAO_ elements (for 4-byte quantities), or else puts
;  226          0323   1 !          the data into the AUX_ buffers (for strings) and puts the final
;  227          0324   1 !          length into the FAO_ buffer.
;  228          0325   1 !
;  229          0326   1 COMPILETIME list_length = 0;
;  230          0327   1
;  231          0328   1 MACRO
;  232        M 0329   1     find_length [item, length] =
;  233          0330   1         %ASSIGN(list_length, list_length + %NUMBER(length))%,
;  234          0331   1
;  235        M 0332   1     bind_names (prefix) [item, length] =
;  236        M 0333   1         %NAME(prefix, item) = %NAME(prefix, 'list')[%NUMBER(list_length)/4]
;  237          0334   1         %ASSIGN(list_length, list_length + %NUMBER(length))%,
;  238          0335   1
;  239        M 0336   1     define_fao_items (list) =
;  240        M 0337   1         %ASSIGN(list_length, 0)
;  241        M 0338   1         find_length(%EXPAND list, %REMAINING)
;  242        M 0339   1         OWN fao_list : VECTOR[%NUMBER(list_length)/4];
;  243        M 0340   1         %ASSIGN(list_length, 0)
;  244          0341   1         BIND bind_names(fao_, %EXPAND list, %REMAINING);%,
;  245          0342   1
;  246        M 0343   1     define_auxiliary_items (list) =
;  247        M 0344   1         %ASSIGN(list_length, 0)
;  248        M 0345   1         find_length(%EXPAND list, %REMAINING)
;  249        M 0346   1         OWN aux_list : VECTOR[%NUMBER(list_length)/4];
;  250        M 0347   1         %ASSIGN(list_length, 0)
;  251          0348   1         BIND bind_names(aux_, %EXPAND list, %REMAINING);%,
;  252          0349   1
;  253        M 0350   1        set_jpi [item, length, buffer, reslen] =
;  254        M 0351   1            %NAME('jpi$_',item)^16 + length,
;  255        M 0352   1            buffer,
;  256          0353   1            reslen%,
;  257          0354   1
;  258        M 0355   1        define jpi_list (list) =
;  259        M 0356   1            OWN jpi_list : VECTOR[3 * (%LENGTH/4) + 1]
;  260          0357   1                INITIAL (set_jpi(%EXPAND list, %REMAINING), 0);%;
;  261          0358   1
;  263          0359  1 !
;  264          0360  1 ! Define the FAO list.  This list is ordered, so that all the data for
;  265          0361  1 ! a particular call to SHOW$WRITE_LINE occurs sequentially.  That way,
;  266          0362  1 ! an argument list is not needed; instead, the address of the first
;  267          0363  1 ! piece of data is given, and the rest just naturally follow.
;  268          0364  1 !
;  269        P 0365  1 define_fao_items(
;  270        P 0366  1
;  271        P 0367  1 ! Header information
;  272        P 0368  1          systime,            4,              ! Dummy (for current date)
;  273        P 0369  1          terminal,           8,              ! terminal name
;  274        P 0370  1          username,           8,              ! username
;  275        P 0371  1 ! Default information
;  276        P 0372  1          pid,                4,              ! process ID
;  277        P 0373  1          prcnam,             8,              ! process name
;  278        P 0374  1          uic,                4,              ! UIC
;  279        P 0375  1          prib,               4,              ! base priority
;  280        P 0376  1          defdev,             8,              ! default device string
;  281        P 0377  1          defdir,             4,              ! default directory
;  282        P 0378  1 ! Quota information
;  283        P 0379  1          account,            8,              ! account name
;  284        P 0380  1          cpulim,             8,              ! cpu limit
;  285        P 0381  1          diolm,              4,              ! direct i/o limit
;  286        P 0382  1          bytcnt,             4,              ! byte count limit
;  287        P 0383  1          biolm,              4,              ! buffered i/o limit
;  288        P 0384  1          tqcnt,              4,              ! timer que entry limit
;  289        P 0385  1          filcnt,             4,              ! open file limit
;  290        P 0386  1          pagfilcnt,          4,              ! paging file quota
;  291        P 0387  1          prclm,              4,              ! subprocess quota
;  292        P 0388  1          dfpfc,              4,              ! default page fault cluster
;  293        P 0389  1          astcnt,             4,              ! ast quota
;  294        P 0390  1          enqcnt,             4,              ! enque limit
;  295        P 0391  1          shrfillm,           4,              ! shared file limit
;  296        P 0392  1          maxdetach,          4,              ! maximum num of detached processes
;  297        P 0393  1          maxjobs,            4,              ! maximum num of active jobs
;  298        P 0394  1 ! Accounting information
;  299        P 0395  1          bufio,              4,              ! buffered i/o count
;  300        P 0396  1          wspeak,             4,              ! peak working set
;  301        P 0397  1          dirio,              4,              ! direct i/o count
;  302        P 0398  1          virtpeak,           4,              ! virtual memory peak
;  303        P 0399  1          pageflts,           4,              ! page fault count
;  304        P 0400  1          volumes,            4,              ! count of mounted volumes
;  305        P 0401  1          imagecount,         4,              ! Count of images executed
;  306        P 0402  1          cputim,             4,              ! cpu time -- later points to quad cputim
;  307          0403  1          logintim,           8);             ! login time (quadword)
;  308          0404  1
;  309          0405  1 !
;  310          0406  1 ! Define the auxiliary buffers.  This includes any strings that are returned,
;  311          0407  1 ! as well as the process privileges.
;  312          0408  1 !
;  313        P 0409  1 define_auxiliary_items(
;  314        P 0410  1          terminal,           16,             ! terminal name
;  315        P 0411  1          username,           16,             ! username
;  316        P 0412  1          prcnam,             16,             ! process name
;  317        P 0413  1          defdev,             256,            ! default device string
;  318        P 0414  1          account,            16,             ! account name
;  319        P 0415  1          cpulim,             16,             ! cpu limit
;  320      P 0416   1              jobprccnt,        4,             ! total process count
;  321      P 0417   1              cputim,           8,             ! Quad CPU time
;  322      P 0418   1              logintim,         8,             ! login time (quadword)
;  323        0419   1              procpriv,         8);            ! privilege bits (quadword)
;  324        0420   1
;  325        0421   1 !
;  326        0422   1 ! Now declare the $GETJPI item list, telling where the data is to go, and
;  327        0423   1 ! where the resultant string lengths should be stored.
;  328        0424   1 !
;  329      P 0425   1 define_jpi_list(
;  330      P 0426   1              terminal,     8,       aux_terminal,    fao_terminal,
;  331      P 0427   1              username,     12,      aux_username,    fao_username,
;  332      P 0428   1              pid,          4,       fao_pid,         0,
;  333      P 0429   1              prcnam,       16,      aux_prcnam,      fao_prcnam,
;  334      P 0430   1              uic,          4,       fao_uic,         0,
;  335      P 0431   1              prib,         1,       fao_prib,        0,
;  336      P 0432   1              account,      8,       aux_account,     fao_account,
;  337      P 0433   1              cpulim,       4,       fao_cpulim,      0,
;  338      P 0434   1              diolm,        2,       fao_diolm,       0,
;  339      P 0435   1              bytcnt,       4,       fao_bytcnt,      0,
;  340      P 0436   1              biolm,        2,       fao_biolm,       0,
;  341      P 0437   1              tqcnt,        2,       fao_tqcnt,       0,
;  342      P 0438   1              filcnt,       2,       fao_filcnt,      0,
;  343      P 0439   1              pagfilcnt,    4,       fao_pagfilcnt,   0,
;  344      P 0440   1              prclm,        2,       fao_prclm,       0,
;  345      P 0441   1              dfpfc,        1,       fao_dfpfc,       0,
;  346      P 0442   1              astcnt,       2,       fao_astcnt,      0,
;  347      P 0443   1              enqcnt,       2,       fao_enqcnt,      0,
;  348      P 0444   1              shrfillm,     2,       fao_shrfillm,    0,
;  349      P 0445   1              maxdetach,    2,       fao_maxdetach,   0,
;  350      P 0446   1              maxjobs,      2,       fao_maxjobs,     0,
;  351      P 0447   1              jobprccnt,    2,       aux_jobprccnt,   0,
;  352      P 0448   1              bufio,        4,       fao_bufio,       0,
;  353      P 0449   1              wspeak,       4,       fao_wspeak,      0,
;  354      P 0450   1              dirio,        4,       fao_dirio,       0,
;  355      P 0451   1              virtpeak,     4,       fao_virtpeak,    0,
;  356      P 0452   1              pageflts,     4,       fao_pageflts,    0,
;  357      P 0453   1              volumes,      4,       fao_volumes,     0,
;  358      P 0454   1              imagecount,   4,       fao_imagecount,  0,
;  359      P 0455   1              cputim,       4,       fao_cputim,      0,
;  360      P 0456   1              logintim,     8,       aux_logintim,    0,
;  361        0457   1              procpriv,     8,       aux_procpriv,    0);
;  363          0458  1 !
;  364          0459  1 ! Make a table of all known privileges, containing the privilege name and
;  365          0460  1 ! text describing it.
;  366          0461  1 !
;  367          0462  1 ! ***** THE PRIVILEGES MUST BE IN BIT NUMBER ORDER *****
;  368          0463  1 !
;  369        P 0464  1 make_priv_table (priv,
;  370        P 0465  1         cmkrnl,                 'may change mode to kernel',
;  371        P 0466  1         cmexec,                 'may change mode to exec',
;  372        P 0467  1         sysnam,                 'may insert in system logical name table',
;  373        P 0468  1         grpnam,                 'may insert in group logical name table',
;  374        P 0469  1         allspool,               'may allocate spooled device',
;  375        P 0470  1         detach,                 'may create detached processes',
;  376        P 0471  1         diagnose,               'may diagnose devices',
;  377        P 0472  1         log_io,                 'may do logical i/o',
;  378        P 0473  1         group,                  'may affect other processes in same group',
;  379        P 0474  1         acnt,                   'may suppress accounting message',
;  380        P 0475  1         prmceb,                 'may create permanent common event clusters',
;  381        P 0476  1         prmmbx,                 'may create permanent mailbox',
;  382        P 0477  1         pswapm,                 'may change process swap mode',
;  383        P 0478  1         altpri,                 'may set any priority value',
;  384        P 0479  1         setprv,                 'may set any privilege bit',
;  385        P 0480  1         tmpmbx,                 'may create temporary mailbox',
;  386        P 0481  1         world,                  'may affect other processes in the world',
;  387        P 0482  1         mount,                  'may execute mount acp function',
;  388        P 0483  1         oper,                   'operator privilege',
;  389        P 0484  1         exquota,                'may exceede quota',
;  390        P 0485  1         netmbx,                 'may create network device',
;  391        P 0486  1         volpro,                 'may override volume protection',
;  392        P 0487  1         phy_io,                 'may do physical i/o',
;  393        P 0488  1         bugchk,                 'may make bug check log entries',
;  394        P 0489  1         prmgbl,                 'may create permanent global sections',
;  395        P 0490  1         sysgbl,                 'may create system wide global sections',
;  396        P 0491  1         pfnmap,                 'may map to specific physical pages',
;  397        P 0492  1         shmem,                  'may create/delete objects in shared memory',
;  398        P 0493  1         sysprv,                 'may access objects via system protection',
;  399        P 0494  1         bypass,                 'bypasses UIC checking',
;  400        P 0495  1         syslck,                 'may lock system wide resources',
;  401        P 0496  1         share,                  'may assign channels to non-shared device',
;  402        P 0497  1         upgrade,                'may upgrade classification',
;  403        P 0498  1         downgrade,              'may downgrade classification',
;  404        P 0499  1         grpprv,                 'group access via system protection',
;  405        P 0500  1         readall,                'may read anything as the owner',
;  406        P 0501  1         tmpjnl,                 'may create temporary journals',
;  407        P 0502  1         prmjnl,                 'may create permanent journals',
;  408          0503  1         security,               'may perform security functions');
;  409          0504  1
;  410          0505  1 !
;  411          0506  1 ! The following GLOBAL declarations are a temporary means of incorporating
;  412          0507  1 ! SHOW PROCESS/CONTINUOUS (aka INFO) into this version of SHOW.  It is not
;  413          0508  1 ! my intention for it to continue in this way. (Famous last words)
;  414          0509  1 !
;  415          0510  1 GLOBAL
;  416          0511  1     proc_a_desc : $BBLOCK[dsc$c_s_bln],
;  417          0512  1     proc_z_name : VECTOR[15,BYTE],
;  418          0513  1     proc_l_pid;
```

```
1 !
1 ! Table of contents
1 !
1
1 FORWARD ROUTINE
1     show$process : NOVALUE,
1     check_privilege : NOVALUE,
1     display_data : NOVALUE,
1     display_tree : NOVALUE,
1     make_tree,
1     next_process : NOVALUE,
1     get_devall,
1     get_devmoun,
1     display_rights : NOVALUE,
1     get_rights_size,
1     get_rights;
1
1 EXTERNAL ROUTINE
1     proc_cont_display : NOVALUE,
1     cli$present,
1     cli$get_value,
1     exe$epid_to_ipid : CVTPID ADDRESSING_MODE (GENERAL),
1     exe$epid_to_pcb : CVTPID ADDRESSING_MODE (GENERAL),
1     lib$get_vm,
1     lib$free_vm,
1     lib$cvt_htb,
1     sch$iolockr : IOLOCK,
1     sch$iounlock : IOLOCK,
1     ioc$scan_iodb : IOSCAN,
1     ioc$cvt_devnam : CVTDEV,
1     show$prcallreg,
1     show$write_line : NOVALUE;
1
1 EXTERNAL
1     ctl$gl_pcb,
1     sch$gl_curpcb,
1     ioc$gl_devlist,
1     scs$ga_localsb,
1     pio$gt_ddstring,
1     sch$gl_maxpix,
1     sch$gl_pixwidth,
1     sch$gl_pcbvec : REF VECTOR;
1
1 EXTERNAL LITERAL
1     sch$c_swppix : UNSIGNED (6);        ! A short literal for the swapper pix
1
1 BUILTIN SUBM;
```

```
  1 GLOBAL ROUTINE show$process : NOVALUE =
  2 BEGIN
  2
  2 !---
  2 !
  2 ! This is the main routine for the SHOW PROCESS function.  All the command
  2 ! qualifiers are gathered, and a $GETJPIW is issued to get the information.
  2 ! If the /CONT qualifier is invoked, the con.rol is transferred to that
  2 ! portion of SHOW.  Otherwise, call the data-display routine.
  2 !
  2 !---
  2
  2 LOCAL
  2     ourpid,                               ! This process's PID
  2     pid,                                  ! PID of requested process
  2     scratch : REF VECTOR,                 ! Scratch area
  2     flags : $BBLOCK[4],                   ! Flags longword
  2     procname : $BBLOCK[dsc$c_s_bln];      ! Process descriptor
  2
  2 ! Check to make sure that the user has the correct privileges to run
  2 ! this image.
  2 !
  2 check_privilege ();
  2
  2 !
  2 ! Collect the qualifiers.  If no qualifiers were present, then show that only
  2 ! the default stuff should be displayed.
  2 !
  3 flags[proc$v_def] = NOT(
  4         (flags[proc$v_quot]  = cli$present(%ASCID 'QUOTAS'))
  4      OR (flags[proc$v_acc]   = cli$present(%ASCID 'ACCOUNTING'))
  4      OR (flags[proc$v_priv]  = cli$present(%ASCID 'PRIVILEGES'))
  4      OR (flags[proc$v_mem]   = cli$present(%ASCID 'MEMORY'))
  4      OR (flags[proc$v_sub]   = cli$present(%ASCID 'SUBPROCESSES'))
  2      OR (flags[proc$v_cont]  = cli$present(%ASCID 'CONTINUOUS')));
  2
  2 !
  2 ! Get the current process PID.
  2 !
  3 BEGIN
  3 LOCAL
  3     status,
  3     iosb : VECTOR[4,WORD],
  3     list : $BBLOCK[16];
  3 list[ 0, 0, 16, 0] = 4;
  3 list[ 2, 0, 16, 0] = jpi$_pid;
  3 list[ 4, 0, 32, 0] = ourpid;
  3 list[ 8, 0, 32, 0] = 0;
  3 list[12, 0, 32, 0] = 0;
P 4 IF (status = $GETJPIW(ITMLST = list,
  4                       IOSB = iosb))
  3 THEN status = .iosb[0];
  3 IF NOT .status
  3 THEN (SIGNAL(.status); RETURN);
  3
  3 !
  3 ! If a process name was specified, convert it to a PID.
```

```
   3 !
   3 pid = 0;                                  ! No PID yet.
   3 $init_dyndesc(procname);                  ! Set up a dynamic descriptor
   3 IF cli$get_value(%ASCID 'PROCESS', procname)
   3 THEN
   4     BEGIN
   4     list[ 4, 0, 32, 0] = pid;
P  5     IF (status = $GETJPIW(ITMLST = list,
P  5                           PRCNAM = procname,
   5                           IOSB   = iosb))
   4     THEN status = .iosb[0];
   4     IF NOT .status
   4     THEN (SIGNAL(.status); RETURN);
   3     END;
   2 END;
   2
   2 !
   2 ! If no process name, then check for a PID.  If the PID was specified,
   2 !convert the ASCII representation to a number.
   2 !
   2 IF cli$get_value(%ASCID 'IDENTIFICATION', procname)
   2 THEN
   3     BEGIN
   3     LOCAL
   3         iosb : VECTOR[4,WORD],
   3         list : $BBLOCK[16],
   3         status;
   3
   4     IF NOT (status = lib$cvt_htb(.procname[dsc$w_length],
   4                                  .procname[dsc$a_pointer],
   4                                  pid))
   3     THEN
   4         BEGIN
   4         SIGNAL(show$_invquaval,         ! Qualifier invalid
   4                2,                       ! with 2 FAO params:
   4                procname,                ! the ID value given,
   4                %ASCID 'IDENTIFICATION'); ! and the /ID qualifier
   4         RETURN;
   3         END;
   3
   3     ! Make sure we get the extended pid, in case the user has used the
   3     ! short form.
   3     !
   3
   3     list[ 0, 0, 16, 0] = 4;
   3     list[ 2, 0, 16, 0] = jpi$_pid;
   3     list[ 4, 0, 32, 0] = pid;
   3     list[ 8, 0, 32, 0] = 0;
   3     list[12, 0, 32, 0] = 0;
   3
P  4     IF (status = $GETJPIW(ITMLST = list,
P  4                           PIDADR = pid,
   4                           IOSB = iosb))
   3     THEN status = .iosb[0];
   3
   3     IF NOT .status
   3     THEN
```

```
  4          BEGIN
  4          SIGNAL(.status);
  4          RETURN;
  3          END;
  2      END;
  2
  2  !
  2  ! If PID is still zero, then no process was specified, i.e. the
  2  ! process in question is the current process.  So, use the current
  2  ! process PID.
  2
  2  IF .pid EQL 0
  2  THEN pid = .ourpid;
  2
  2  !
  2  ! If the /ALL qualifier is present, then turn on the proper bits.
  2  ! If the pid is for the current process, then allow memory
  2  ! and subprocesses too.
  2  !
  2  IF cli$present(%ASCID 'ALL')
  2  THEN
  3      BEGIN
  3      flags[proc$v_def]   = flags[proc$v_quot]
  3                          = flags[proc$v_acc]
  3                          = flags[proc$v_priv]
  3                          = true;
  3
  3      IF .pid EQL .ourpid
  3      THEN
  3      flags[proc$v_mem]   = flags[proc$v_sub]
  3                          = true;
  2      END;
  2
  2  !
  2  ! Now for some further checks.  If /MEMORY or /SUBPROCESSES was
  2  ! specified, and the requested process is not the current process,
  2  ! then signal that it can't be done.
  2  !
  2  IF .ourpid NEQ .pid
  3  AND (.flags[proc$v_mem] OR .flags[proc$v_sub])
  2  THEN (SIGNAL(show$_confqual); RETURN);
  2
  2  !
  2  ! Obtain the data for the requested process.
  2  !
  3  BEGIN
  3  LOCAL
  3      status,
  3      iosb : VECTOR[4,WORD];
P 4  IF (status = $GETJPIW(PIDADR = pid,
P 4                        ITMLST = jpi_list,
  4                        IOSB = iosb))
  3  THEN status = .iosb[0];
  3  IF NOT .status
  3  THEN (SIGNAL(.status); RETURN);
  3  pid = .fao_pid;
  2  END;
```

```
;  639          0732  2
;  640          0733  2  !
;  641          0734  2  ! If /CONTINUOUS was specified, then transfer control
;  642          0735  2  !
;  643          0736  2  IF .flags[proc$v_cont]
;  644          0737  2  THEN
;  645          0738  3      BEGIN
;  646          0739  3      proc_l_pid = .fao_pid;
;  647          0740  3      proc_a_desc[dsc$w_length] = fao_prcnam;
;  648          0741  3      proc_a_desc[dsc$a_pointer] = aux_prcnam;
;  649          0742  3      CH$MOVE(.fao_prcnam, aux_prcnam, proc_z_name);
;  650          0743  3      proc_cont_display();
;  651          0744  3      RETURN;
;  652          0745  2      END;
;  653          0746  2
;  654          0747  2  !
;  655          0748  2  ! Grab a large chunk of memory in which to put data.  The present
;  656          0749  2  ! ''algorithm'' is to grab 64 pages, which should be more than enough,
;  657          0750  2  ! at least it has been so far.
;  658          0751  2  !
;  659          0752  3  BEGIN
;  660          0753  3  REGISTER status;
;  661          0754  4  IF NOT (status = lib$get_vm(%REF(64*512), scratch))
;  662          0755  3  THEN (SIGNAL(.status); RETURN);
;  663          0756  3  scratch[0] = 64*512;                    ! Put the size of the scratch area
;  664          0757  3                                          ! in the first longword
;  665          0758  2  END;
;  666          0759  2
;  667          0760  2
;  668          0761  2  !
;  669          0762  2  ! Now to print all the stuff.
;  670          0763  2  !
;  671          0764  2  display_data (.scratch, flags, .ourpid);
;  672          0765  2
;  673          0766  2  RETURN;
;  674          0767  1  END;
;  INFO#250             L1:0687
; Referenced LOCAL symbol OURPID is probably not initialized


                                                                                  .TITLE  SHOWPROCESS
                                                                                  .IDENT  \V04-000\

                                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                                  00  4C  4E  52  4B  4D  43  06  00000 P.AAA:   .ASCII  <6>\CMKRNL\<0>
64  6F  6D  20  65  67  6E  61  68  63  20  79  61  6D  19  00008 P.AAB:   .ASCII  <25>\may change mode to kernel\<0><0>
        00  00  6C  65  6E  72  65  6B  20  6F  74  20  65  00017
                                  00  43  45  58  45  4D  43  06  00024 P.AAC:   .ASCII  <6>\CMEXEC\<0>
64  6F  6D  20  65  67  6E  61  68  63  20  79  61  6D  17  0002C P.AAD:   .ASCII  <23>\may change mode to exec\
                            63  65  78  65  20  6F  74  20  65  0003B
                                  00  4D  41  4E  53  59  53  06  00044 P.AAE:   .ASCII  <6>\SYSNAM\<0>
20  6E  69  20  74  72  65  73  6E  69  20  79  61  6D  27  0004C P.AAF:   .ASCII  \'may insert in system logical name table\
20  6C  61  63  69  67  6F  6C  20  6D  65  74  73  79  73  0005B
                        65  6C  62  61  74  20  65  6D  61  6E  0006A
                                  00  4D  41  4E  50  52  47  06  00074 P.AAG:   .ASCII  <6>\GRPNAM\<0>
20  6E  69  20  74  72  65  73  6E  69  20  79  61  6D  26  0007C P.AAH:   .ASCII  \&may insert in group logical name table-
```

```
6E  20  6C  61  63  69  67  6F  6C  20  70  75  6F  72  67  0008B                    \<0>
                    00  65  6C  62  61  74  20  65  6D  61  0009A
            00  00  00  4C  4F  4F  50  53  4C  4C  41  08  000A4 P.AAI:   .ASCII    <8>\ALLSPOOL\<0><0><0>
73  20  65  74  61  63  6F  6C  6C  61  20  79  61  6D  1B  000B0 P.AAJ:   .ASCII    <27>\may allocate spooled device\
        65  63  69  76  65  64  20  64  65  6C  6F  6F  70  000BF
                            00  48  43  41  54  45  44  06  000CC P.AAK:   .ASCII    <6>\DETACH\<0>
74  65  64  20  65  74  61  65  72  63  20  79  61  6D  1D  000D4 P.AAL:   .ASCII    <29>\may create detached processes\<0>
73  65  73  73  65  63  6F  72  70  20  64  65  68  63  61  000E3
                                                        00  000F2
                                                        00  000F3          .ASCII    <0>
            00  00  00  45  53  4F  4E  47  41  49  44  08  000F4 P.AAM:   .ASCII    <8>\DIAGNOSE\<0><0><0>
64  20  65  73  6F  6E  67  61  69  64  20  79  61  6D  14  00100 P.AAN:   .ASCII    <20>\may diagnose devices\<0><0><0>
                        00  00  00  73  65  63  69  76  65  0010F
                            00  4F  49  5F  47  4F  4C  06  00118 P.AAO:   .ASCII    <6>\LOG_IO\<0>
6C  61  63  69  67  6F  6C  20  6F  64  20  79  61  6D  12  00120 P.AAP:   .ASCII    <18>\may do logical i/o\<0>
                                        00  6F  2F  69  20  0012F
                            00  00  50  55  4F  52  47  05  00134 P.AAQ:   .ASCII    <5>\GROUP\<0><0>
68  74  6F  20  74  63  65  66  66  61  20  79  61  6D  28  0013C P.AAR:   .ASCII    \(may affect other processes in same grou\
6E  69  20  73  65  73  73  65  63  6F  72  70  20  72  65  0014B
                    75  6F  72  67  20  65  6D  61  73  20  0015A
                                            00  00  00  70  00164          .ASCII    \p\<0><0><0>
                            00  00  00  54  4E  43  41  04  00168 P.AAS:   .ASCII    <4>\ACNT\<0><0><0>
61  20  73  73  65  72  70  70  75  73  20  79  61  6D  1F  00170 P.AAT:   .ASCII    <31>\may suppress accounting message\
61  73  73  65  6D  20  67  6E  69  74  6E  75  6F  63  63  0017F
                                                    65  67  0018E
                            00  42  45  43  4D  52  50  06  00190 P.AAU:   .ASCII    <6>\PRMCEB\<0>
72  65  70  20  65  74  61  65  72  63  20  79  61  6D  2A  00198 P.AAV:   .ASCII    \*may create permanent common event clust\
65  20  6E  6F  6D  6D  6F  63  20  74  6E  65  6E  61  6D  001A7
                    74  73  75  6C  63  20  74  6E  65  76  001B6
                                            00  73  72  65  001C0          .ASCII    \ers\<0>
                            00  58  42  4D  4D  52  50  06  001C4 P.AAW:   .ASCII    <6>\PRMMBX\<0>
72  65  70  20  65  74  61  65  72  63  20  79  61  6D  1C  001CC P.AAX:   .ASCII    <28>\may create permanent mailbox\<0>
00  78  6F  62  6C  69  61  6D  20  74  6E  65  6E  61  6D  001DB
                                                    00  00  001EA          .ASCII    <0><0>
                            00  4D  50  41  57  53  50  06  001EC P.AAY:   .ASCII    <6>\PSWAPM\<0>
6F  72  70  20  65  67  6E  61  68  63  20  79  61  6D  1C  001F4 P.AAZ:   .ASCII    <28>\may change process swap mode\<0>
00  65  64  6F  6D  20  70  61  77  73  20  73  73  65  63  00203
                                                    00  00  00212          .ASCII    <0><0>
                            00  49  52  50  54  4C  41  06  00214 P.ABA:   .ASCII    <6>\ALTPRI\<0>
72  70  20  79  6E  61  20  74  65  73  20  79  61  6D  1A  0021C P.ABB:   .ASCII    <26>\may set any priority value\<0>
        00  65  75  6C  61  76  20  79  74  69  72  6F  69  0022B
                            00  56  52  50  54  45  53  06  00238 P.ABC:   .ASCII    <6>\SETPRV\<0>
72  70  20  79  6E  61  20  74  65  73  20  79  61  6D  19  00240 P.ABD:   .ASCII    <25>\may set any privilege bit\<0><0>
        00  00  74  69  62  20  65  67  65  6C  69  76  69  0024F
                            00  58  42  4D  50  4D  54  06  0025C P.ABE:   .ASCII    <6>\TMPMBX\<0>
6D  65  74  20  65  74  61  65  72  63  20  79  61  6D  1C  00264 P.ABF:   .ASCII    <28>\may create temporary mailbox\<0>
00  78  6F  62  6C  69  61  6D  20  79  72  61  72  6F  70  00273
                                                    00  00  00282          .ASCII    <0><0>
                            00  00  44  4C  52  4F  57  05  00284 P.ABG:   .ASCII    <5>\WORLD\<0><0>
68  74  6F  20  74  63  65  66  66  61  20  79  61  6D  27  0028C P.ABH:   .ASCII    \'may affect other processes in the world\
6E  69  20  73  65  73  73  65  63  6F  72  70  20  72  65  0029B
                    64  6C  72  6F  77  20  65  68  74  20  002AA
                            00  00  54  4E  55  4F  4D  05  002B4 P.ABI:   .ASCII    <5>\MOUNT\<0><0>
6F  6D  20  65  74  75  63  65  78  65  20  79  61  6D  1E  002BC P.ABJ:   .ASCII    <30>\may execute mount acp function\<0>
6F  69  74  63  6E  75  66  20  70  63  61  20  74  6E  75  002CB
                                                    00  6E  002DA
                            00  00  00  52  45  50  4F  04  002DC P.ABK:   .ASCII    <4>\OPER\<0><0><0>
```

```
69  76  69  72  70  20  72  6F  74  61  72  65  70  6F  12  002E4 P.ABL:  .ASCII  <18>\operator privilege\<0>
                                        00  65  67  65  6C  002F3
                            41  54  4F  55  51  58  45  07  002F8 P.ABM:  .ASCII  <7>\EXQUOTA\
75  71  20  65  64  65  65  63  78  65  20  79  61  6D  11  00300 P.ABN:  .ASCII  <17>\may exceede quota\<0><0>
                                        00  00  61  74  6F  0030F
                            00  58  42  4D  54  45  4E  06  00314 P.ABO:  .ASCII  <6>\NETMBX\<0>
74  65  6E  20  65  74  61  65  72  63  20  79  61  6D  19  0031C P.ABP:  .ASCII  <25>\may create network device\<0><0>
        00  00  65  63  69  76  65  64  20  6B  72  6F  77  0032B
                            00  4F  52  50  4C  4F  56  06  00338 P.ABQ:  .ASCII  <6>\VOLPRO\<0>
76  20  65  64  69  72  72  65  76  6F  20  79  61  6D  1E  00340 P.ABR:  .ASCII  <30>\may override volume protection\<0>
6F  69  74  63  65  74  6F  72  70  20  65  6D  75  6C  6F  0034F
                                                    00  6E  0035E
                            00  4F  49  5F  59  48  50  06  00360 P.ABS:  .ASCII  <6>\PHY_IO\<0>
61  63  69  73  79  68  70  20  6F  64  20  79  61  6D  13  00368 P.ABT:  .ASCII  <19>\may do physical i/o\
                                        6F  2F  69  20  6C  00377
                            00  4B  48  43  47  55  42  06  0037C P.ABU:  .ASCII  <6>\BUGCHK\<0>
63  20  67  75  62  20  65  6B  61  6D  20  79  61  6D  1E  00384 P.ABV:  .ASCII  <30>\may make bug check log entries\<0>
65  69  72  74  6E  65  20  67  6F  6C  20  6B  63  65  68  00393
                                                    00  73  003A2
                            00  4C  42  47  4D  52  50  06  003A4 P.ABW:  .ASCII  <6>\PRMGBL\<0>
72  65  70  20  65  74  61  65  72  63  20  79  61  6D  24  003AC P.ABX:  .ASCII  \$may create permanent global sections\<0>
73  20  6C  61  62  6F  6C  67  20  74  6E  65  6E  61  6D  003BB
                            00  73  6E  6F  69  74  63  65  003CA
                                                    00  00  003D2         .ASCII  <0><0>
                            00  4C  42  47  53  59  53  06  003D4 P.ABY:  .ASCII  <6>\SYSGBL\<0>
73  79  73  20  65  74  61  65  72  63  20  79  61  6D  26  003DC P.ABZ:  .ASCII  \&may create system wide global sections-
6C  61  62  6F  6C  67  20  65  64  69  77  20  6D  65  74  003EB                 \<0>
                    00  73  6E  6F  69  74  63  65  73  20  003FA
                            00  50  41  4D  4E  46  50  06  00404 P.ACA:  .ASCII  <6>\PFNMAP\<0>
65  70  73  20  6F  74  20  70  61  6D  20  79  61  6D  22  0040C P.ACB:  .ASCII  \"may map to specific physical pages\<0>
20  6C  61  63  69  73  79  68  70  20  63  69  66  69  63  0041B
                                    00  73  65  67  61  70  0042A
                            00  00  4D  45  4D  48  53  05  00430 P.ACC:  .ASCII  <5>\SHMEM\<0><0>
6C  65  64  2F  65  74  61  65  72  63  20  79  61  6D  2A  00438 P.ACD:  .ASCII  \*may create/delete objects in shared mem\
20  6E  69  20  73  74  63  65  6A  62  6F  20  65  74  65  00447
                    6D  65  6D  20  64  65  72  61  68  73  00456
                                            00  79  72  6F  00460         .ASCII  \ory\<0>
                            00  56  52  50  53  59  53  06  00464 P.ACE:  .ASCII  <6>\SYSPRV\<0>
6A  62  6F  20  73  73  65  63  63  61  20  79  61  6D  28  0046C P.ACF:  .ASCII  \(may access objects via system protectio\
6D  65  74  73  79  73  20  61  69  76  20  73  74  63  65  0047B
                    6F  69  74  63  65  74  6F  72  70  20  0048A
                                            00  00  00  6E  00494         .ASCII  \n\<0><0><0>
                            00  53  53  41  50  59  42  06  00498 P.ACG:  .ASCII  <6>\BYPASS\<0>
63  20  43  49  55  20  73  65  73  73  61  70  79  62  15  004A0 P.ACH:  .ASCII  <21>\bypasses UIC checking\<0><0>
                        00  00  67  6E  69  6B  63  65  68  004AF
                            00  4B  43  4C  53  59  53  06  004B8 P.ACI:  .ASCII  <6>\SYSLCK\<0>
65  74  73  79  73  20  6B  63  6F  6C  20  79  61  6D  1E  004C0 P.ACJ:  .ASCII  <30>\may lock system wide resources\<0>
65  63  72  75  6F  73  65  72  20  65  64  69  77  20  6D  004CF
                                                    00  73  004DE
                            00  00  45  52  41  48  53  05  004E0 P.ACK:  .ASCII  <5>\SHARE\<0><0>
61  68  63  20  6E  67  69  73  73  61  20  79  61  6D  28  004E8 P.ACL:  .ASCII  \(may assign channels to non-shared devic\
68  73  2D  6E  6F  6E  20  6F  74  20  73  6C  65  6E  6E  004F7
                    63  69  76  65  64  20  64  65  72  61  00506
                                            00  00  00  65  00510         .ASCII  \e\<0><0><0>
                            45  44  41  52  47  50  55  07  00514 P.ACM:  .ASCII  <7>\UPGRADE\
6C  63  20  65  64  61  72  67  70  75  20  79  61  6D  1A  0051C P.ACN:  .ASCII  <26>\may upgrade classification\<0>
        00  6E  6F  69  74  61  63  69  66  69  73  73  61  0052B
```

```
            00  00  45  44  41  52  47  4E  57  4F  44  09  00538 P.ACO:  .ASCII  <9>\DOWNGRADE\<0><0>
20  65  64  61  72  67  6E  77  6F  64  20  79  61  6D  1C  00544 P.ACP:  .ASCII  <28>\may downgrade classification\<0>
00  6E  6F  69  74  61  63  69  66  69  73  73  61  6C  63  00553
                                                    00  00  00562         .ASCII  <0><0>
                            00  56  52  50  50  52  47  06  00564 P.ACQ:  .ASCII  <6>\GRPPRV\<0>
76  20  73  73  65  63  63  61  20  70  75  6F  72  67  22  0056C P.ACR:  .ASCII  \"group access via system protection\<0>
65  74  6F  72  70  20  6D  65  74  73  79  73  20  61  69  0057B
                                        00  6E  6F  69  74  63  0058A
                            4C  4C  41  44  41  45  52  07  00590 P.ACS:  .ASCII  <7>\READALL\
68  74  79  6E  61  20  64  61  65  72  20  79  61  6D  1E  00598 P.ACT:  .ASCII  <30>\may read anything as the owner\<0>
65  6E  77  6F  20  65  68  74  20  73  61  20  67  6E  69  005A7
                                                    00  72  005B6
                            00  4C  4E  4A  50  4D  54  06  005B8 P.ACU:  .ASCII  <6>\TMPJNL\<0>
6D  65  74  20  65  74  61  65  72  63  20  79  61  6D  1D  005C0 P.ACV:  .ASCII  <29>\may create temporary journals\<0>
73  6C  61  6E  72  75  6F  6A  20  79  72  61  72  6F  70  005CF
                                                        00  005DE
                                                        00  005DF         .ASCII  <0>
                            00  4C  4E  4A  4D  52  50  06  005E0 P.ACW:  .ASCII  <6>\PRMJNL\<0>
72  65  70  20  65  74  61  65  72  63  20  79  61  6D  1D  005E8 P.ACX:  .ASCII  <29>\may create permanent journals\<0>
73  6C  61  6E  72  75  6F  6A  20  74  6E  65  6E  61  6D  005F7
                                                        00  00606
                                                        00  00607         .ASCII  <0>
            00  00  00  59  54  49  52  55  43  45  53  08  00608 P.ACY:  .ASCII  <8>\SECURITY\<0><0><0>
65  73  20  6D  72  6F  66  72  65  70  20  79  61  6D  1E  00614 P.ACZ:  .ASCII  <30>\may perform security functions\<0>
6E  6F  69  74  63  6E  75  66  20  79  74  69  72  75  63  00623
                                                    00  73  00632
                            00  00  53  41  54  4F  55  51  00634 P.ADB:  .ASCII  \QUOTAS\<0><0>
                                                  010E0006  0063C P.ADA:  .LONG   17694726
                                                  00000000' 00640         .ADDRESS P.ADB
            00  00  47  4E  49  54  4E  55  4F  43  43  41  00644 P.ADD:  .ASCII  \ACCOUNTING\<0><0>
                                                  010E000A  00650 P.ADC:  .LONG   17694730
                                                  00000000' 00654         .ADDRESS P.ADD
            00  00  53  45  47  45  4C  49  56  49  52  50  00658 P.ADF:  .ASCII  \PRIVILEGES\<0><0>
                                                  010E000A  00664 P.ADE:  .LONG   17694730
                                                  00000000' 00668         .ADDRESS P.ADF
                            00  00  59  52  4F  4D  45  4D  0066C P.ADH:  .ASCII  \MEMORY\<0><0>
                                                  010E0006  00674 P.ADG:  .LONG   17694726
                                                  00000000' 00678         .ADDRESS P.ADH
            53  45  53  53  45  43  4F  52  50  42  55  53  0067C P.ADJ:  .ASCII  \SUBPROCESSES\
                                                  010E000C  00688 P.ADI:  .LONG   17694732
                                                  00000000' 0068C         .ADDRESS P.ADJ
            00  00  53  55  4F  55  4E  49  54  4E  4F  43  00690 P.ADL:  .ASCII  \CONTINUOUS\<0><0>
                                                  010E000A  0069C P.ADK:  .LONG   17694730
                                                  00000000' 006A0         .ADDRESS P.ADL
                            00  53  53  45  43  4F  52  50  006A4 P.ADN:  .ASCII  \PROCESS\<0>
                                                  010E0007  006AC P.ADM:  .LONG   17694727
                                                  00000000' 006B0         .ADDRESS P.ADN
00  4E  4F  49  54  41  43  49  46  49  54  4E  45  44  49  006B4 P.ADP:  .ASCII  \IDENTIFICATION\<0><0>
                                                        00  006C3
                                                  010E000E  006C4 P.ADO:  .LONG   17694734
                                                  00000000' 006C8         .ADDRESS P.ADP
00  4E  4F  49  54  41  43  49  46  49  54  4E  45  44  49  006CC P.ADR:  .ASCII  \IDENTIFICATION\<0><0>
                                                        00  006DB
                                                  010E000E  006DC P.ADQ:  .LONG   17694734
                                                  00000000' 006E0         .ADDRESS P.ADR
                                            00  4C  4C  41  006E4 P.ADT:  .ASCII  \ALL\<0>
                                                  010E0003  006E8 P.ADS:  .LONG   17694723
```

```
                 00000000' 006EC          .ADDRESS P.ADT                                            ;

                                          .PSECT  $OWN$,NOEXE,2

                           00000 FAO_LIST:
                                          .BLKB   160
                           000A0 AUX_LIST:
                                          .BLKB   364
                 031D0008  0020C JPI_LIST:
                                          .LONG   52232200                                          ;
      00000000'  00000000' 00210          .ADDRESS AUX_TERMINAL, FAO_TERMINAL                       ;
                 0202000C  00218          .LONG   33685516                                          ;
      00000000'  00000000' 0021C          .ADDRESS AUX_USERNAME, FAO_USERNAME                       ;
                 03190004  00224          .LONG   51970052                                          ;
                 00000000' 00228          .ADDRESS FAO_PID                                          ;
      031C0010   00000000  0022C          .LONG   0, 52166672                                       ;
      00000000'  00000000' 00234          .ADDRESS AUX_PRCNAM, FAO_PRCNAM                           ;
                 03040004  0023C          .LONG   50593796                                          ;
                 00000000' 00240          .ADDRESS FAO_UIC                                          ;
      03090001   00000000  00244          .LONG   0, 50921473                                       ;
                 00000000' 0024C          .ADDRESS FAO_PRIB                                         ;
      02030008   00000000  00250          .LONG   0, 33751048                                       ;
      00000000'  00000000' 00258          .ADDRESS AUX_ACCOUNT, FAO_ACCOUNT                         ;
                 040D0004  00260          .LONG   67960836                                          ;
                 00000000' 00264          .ADDRESS FAO_CPULIM                                       ;
      03130002   00000000  00268          .LONG   0, 51576834                                       ;
                 00000000' 00270          .ADDRESS FAO_DIOLM                                        ;
      03110004   00000000  00274          .LONG   0, 51445764                                       ;
                 00000000' 0027C          .ADDRESS FAO_BYTCNT                                       ;
      03100002   00000000  00280          .LONG   0, 51380226                                       ;
                 00000000' 00288          .ADDRESS FAO_BIOLM                                        ;
      03150002   00000000  0028C          .LONG   0, 51707906                                       ;
                 00000000' 00294          .ADDRESS FAO_TQCNT                                        ;
      03140002   00000000  00298          .LONG   0, 51642370                                       ;
                 00000000' 002A0          .ADDRESS FAO_FILCNT                                       ;
      04140004   00000000  002A4          .LONG   0, 68419588                                       ;
                 00000000' 002AC          .ADDRESS FAO_PAGFILCNT                                    ;
      04080002   00000000  002B0          .LONG   0, 67633154                                       ;
                 00000000' 002B8          .ADDRESS FAO_PRCLM                                        ;
      04060001   00000000  002BC          .LONG   0, 67502081                                       ;
                 00000000' 002C4          .ADDRESS FAO_DFPFC                                        ;
      030E0002   00000000  002C8          .LONG   0, 51249154                                       ;
                 00000000' 002D0          .ADDRESS FAO_ASTCNT                                       ;
      031F0002   00000000  002D4          .LONG   0, 52363266                                       ;
                 00000000' 002DC          .ADDRESS FAO_ENQCNT                                       ;
      02100002   00000000  002E0          .LONG   0, 34603010                                       ;
                 00000000' 002E8          .ADDRESS FAO_SHRFILLM                                     ;
      020E0002   00000000  002EC          .LONG   0, 34471938                                       ;
                 00000000' 002F4          .ADDRESS FAO_MAXDETACH                                    ;
      020F0002   00000000  002F8          .LONG   0, 34537474                                       ;
                 00000000' 00300          .ADDRESS FAO_MAXJOBS                                      ;
      031E0002   00000000  00304          .LONG   0, 52297730                                       ;
                 00000000' 0030C          .ADDRESS AUX_JOBPRCCNT                                    ;
      040C0004   00000000  00310          .LONG   0, 67895300                                       ;
                 00000000' 00318          .ADDRESS FAO_BUFIO                                        ;
      02010004   00000000  0031C          .LONG   0, 33619972                                       ;
                 00000000' 00324          .ADDRESS FAO_WSPEAK                                       ;
```

```
                                                  040B0004  00000000  00328          .LONG    0, 67829764                                 ;
                                                            00000000' 00330          .ADDRESS FAO_DIRIO                                   ;
                                                  02000004  00000000  00334          .LONG    0, 33554436                                 ;
                                                            00000000' 0033C          .ADDRESS FAO_VIRTPEAK                                ;
                                                  040A0004  00000000  00340          .LONG    0, 67764228                                 ;
                                                            00000000' 00348          .ADDRESS FAO_PAGEFLTS                                ;
                                                  02050004  00000000  0034C          .LONG    0, 33882116                                 ;
                                                            00000000' 00354          .ADDRESS FAO_VOLUMES                                 ;
                                                  041A0004  00000000  00358          .LONG    0, 68812804                                 ;
                                                            00000000' 00360          .ADDRESS FAO_IMAGECOUNT                              ;
                                                  04070004  00000000  00364          .LONG    0, 67567620                                 ;
                                                            00000000' 0036C          .ADDRESS FAO_CPUTIM                                  ;
                                                  02060008  00000000  00370          .LONG    0, 33947656                                 ;
                                                            00000000' 00378          .ADDRESS AUX_LOGINTIM                                ;
                                                  02040008  00000000  0037C          .LONG    0, 33816584                                 ;
                                                            00000000' 00384          .ADDRESS AUX_PROCPRIV                                ;
                                                  00000000  00000000  00388          .LONG    0, 0                                        ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00390 PRIV_TABLE:
                                                                     .ADDRESS P.AAA, P.AAB, P.AAC, P.AAD, P.AAE, -          ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 003A8             P.AAF, P.AAG, P.AAH, P.AAI, P.AAJ, P.AAK, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 003CC             P.AAL, P.AAM, P.AAN, P.AAO, P.AAP, P.AAQ, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 003D8             P.AAR, P.AAS, P.AAT, P.AAU, P.AAV, P.AAW, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 003F0             P.AAX, P.AAY, P.AAZ, P.ABA, P.ABB, P.ABC, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00408             P.ABD, P.ABE, P.ABF, P.ABG, P.ABH, P.ABI, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00420             P.ABJ, P.ABK, P.ABL, P.ABM, P.ABN, P.ABO, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00438             P.ABP, P.ABQ, P.ABR, P.ABS, P.ABT, P.ABU, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00450             P.ABV, P.ABW, P.ABX, P.ABY, P.ABZ, P.ACA, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00468             P.ACB, P.ACC, P.ACD, P.ACE, P.ACF, P.ACG, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00480             P.ACH, P.ACI, P.ACJ, P.ACK, P.ACL, P.ACM, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00498             P.ACN, P.ACO, P.ACP, P.ACQ, P.ACR, P.ACS, -   ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 004B0             P.ACT, P.ACU, P.ACV, P.ACW, P.ACX, P.ACY, -   ;
                                                                              P.ACZ                                         ;
                                                            004C8          .BLKB    4

                                                                           .PSECT   $GLOBAL$,NOEXE,2

                                                            00000 PROC_A_DESC::
                                                                           .BLKB    8
                                                            00008 PROC_Z_NAME::
                                                                           .BLKB    15
                                                            00017          .BLKB    1
                                                            00018 PROC_L_PID::
                                                                           .BLKB    4

                                                                  FAO_SYSTIME=        FAO_LIST
                                                                  FAO_TERMINAL=       FAO_LIST+4
                                                                  FAO_USERNAME=       FAO_LIST+12
                                                                  FAO_PID=            FAO_LIST+20
                                                                  FAO_PRCNAM=         FAO_LIST+24
                                                                  FAO_UIC=            FAO_LIST+32
                                                                  FAO_PRIB=           FAO_LIST+36
                                                                  FAO_DEFDEV=         FAO_LIST+40
                                                                  FAO_DEFDIR=         FAO_LIST+48
                                                                  FAO_ACCOUNT=        FAO_LIST+52
                                                                  FAO_CPULIM=         FAO_LIST+60
                                                                  FAO_DIOLM=          FAO_LIST+68
                                                                  FAO_BYTCNT=         FAO_LIST+72
```

```
                                                    FAO_BIOLM=              FAO_LIST+76
                                                    FAO_TQCNT=              FAO_LIST+80
                                                    FAO_FILCNT=             FAO_LIST+84
                                                    FAO_PAGFILCNT=          FAO_LIST+88
                                                    FAO_PRCLM=              FAO_LIST+92
                                                    FAO_DFPFC=              FA^_LIST+96
                                                    FAO_ASTCNT=             FAO_LIST+100
                                                    FAO_ENQCNT=             FAO_LIST+104
                                                    FAO_SHRFILLM=           FAO_LIST+108
                                                    FAO_MAXDETACH=          FAO_LIST+112
                                                    FAO_MAXJOBS=            FAO_LIST+116
                                                    FAO_BUFIO=              FAO_LIST+120
                                                    FAO_WSPEAK=             FAO_LIST+124
                                                    FAO_DIRIO=              FAO_LIST+128
                                                    FAO_VIRTPEAK=           FAO_LIST+132
                                                    FAO_PAGEFLTS=           FAO_LIST+136
                                                    FAO_VOLUMES=            FAO_LIST+140
                                                    FAO_IMAGECOUNT=         FAO_LIST+144
                                                    FAO_CPUTIM=             FAO_LIST+148
                                                    FAO_LOGINTIM=           FAO_LIST+152
                                                    AUX_TERMINAL=           AUX_LIST
                                                    AUX_USERNA 1E=          AUX_LIST+16
                                                    AUX_PRCNAM=             AUX_LIST+32
                                                    AUX_DEFDEV=             AUX_LIST+48
                                                    AUX_ACCOUNT=            AUX_LIST+304
                                                    AUX_CPULIM=             AUX_LIST+320
                                                    AUX_JOBPRCCNT=          AUX_LIST+336
                                                    AUX_CPUTIM=             AUX_LIST+340
                                                    AUX_LOGINTIM=           AUX_LIST+348
                                                    AUX_PROCPRIV=           AUX_LIST+356
                                                            .EXTRN  PROC_CONT_DISPLAY
                                                            .EXTRN  CLI$PRESENT, CLI$GET_VALUE
                                                            .EXTRN  EXE$EPID_TO_IPID
                                                            .EXTRN  EXE$EPID_TO_PCB
                                                            .EXTRN  LIB$GET_VM, LIB$FREE_VM
                                                            .EXTRN  LIB$CVT_HTB, SCH$IOLOCKR
                                                            .EXTRN  SCH$IOUNLOCK, IOC$SCAN_IODB
                                                            .EXTRN  IOC$CVT_DEVNAM, SHOW$PRCALLREG
                                                            .EXTRN  SHOW$WRITE_LINE
                                                            .EXTRN  CTL$GL_PCB, SCH$GL_CURPCB
                                                            .EXTRN  IOC$GL_DEVLIST, SCS$GA_LOCALSB
                                                            .EXTRN  PIO$GT_DDSTRING
                                                            .EXTRN  SCH$GL_MAXPIX, SCH$GL_PIXWIDTH
                                                            .EXTRN  SCH$GL_PCBVEC, SCH$C_SWPPIX
                                                            .EXTRN  SYS$GETJPIW

                                                            .PSECT  $CODE$,NOWRT,2

                                    OFFC 00000              .ENTRY  SHOW$PROCESS, Save R2,R3,R4,R5,R6,R7,R8,R9,-;  0561
                                                                    R10,R11                                          ;
                         5B 00000000G  00  9E 00002         MOVAB   LIB$SIGNAL, R11                                  ;
                         5A 00000000G  00  9E 00009         MOVAB   CLI$GET_VALUE, R10                               ;
                         59     0000'  CF  9E 00010         MOVAB   FAO_PID, R9                                      ;
                         58 00000000G  00  9E 00015         MOVAB   SYS$GETJPIW, R8                                  ;
                         57     0000'  CF  9E 0001C         MOVAB   P.ADA, R7                                        ;
                         56 00000000G  00  9E 00021         MOVAB   CLI$PRESENT, R6                                  ;
                         5E            34  C2 00028         SUBL2   #52, SP                                          ;
```

```
                                 0000V  CF            00  FB 0002B        CALLS   #0, CHECK_PRIVILEGE                                 ; 0583
                                                      57  DD 00030        PUSHL   R7                                                  ; 0590
                                        66            01  FB 00032        CALLS   #1, CLI$PRESENT                                     ;
                                        52            50  D0 00035        MOVL    R0, R2                                              ;
     10   AE             01             01            52  F0 00038        INSV    R2, #1, #1, FLAGS                                   ;
                                                 14   A7  9F 0003E        PUSHAB  P.ADC                                               ; 0591
                                        66            01  FB 00041        CALLS   #1, CLI$PRESENT                                     ;
     10   AE             01             02            50  F0 00044        INSV    R0, #2, #1, FLAGS                                   ;
                                        52            50  C8 0004A        BISL2   R0, R2                                              ;
                                                 28   A7  9F 0004D        PUSHAB  P.ADE                                               ; 0592
                                        66            01  FB 00050        CALLS   #1, CLI$PRESENT                                     ;
     10   AE             01             03            50  F0 00053        INSV    R0, #3, #1, FLAGS                                   ;
                                        52            50  C8 00059        BISL2   R0, R2                                              ;
                                                 38   A7  9F 0005C        PUSHAB  P.ADG                                               ; 0593
                                        66            01  FB 0005F        CALLS   #1, CLI$PRESENT                                     ;
     10   AE             01             04            50  F0 00062        INSV    R0, #4, #1, FLAGS                                   ;
                                        52            50  C8 00068        BISL2   R0, R2                                              ;
                                                 4C   A7  9F 0006B        PUSHAB  P.ADI                                               ; 0594
                                        66            01  FB 0006E        CALLS   #1, CLI$PRESENT                                     ;
     10   AE             01             05            50  F0 00071        INSV    R0, #5, #1, FLAGS                                   ;
                                        52            50  C8 00077        BISL2   R0, R2                                              ;
                                                 60   A7  9F 0007A        PUSHAB  P.ADK                                               ; 0595
                                        66            01  FB 0007D        CALLS   #1, CLI$PRESENT                                     ;
     11   AE             01             00            50  F0 00080        INSV    R0, #0, #1, FLAGS+1                                 ;
                                        52            50  C8 00086        BISL2   R0, R2                                              ;
                                        50            52  D2 00089        MCOML   R2, R0                                              ; 0589
     10   AE             01             00            50  F0 0008C        INSV    R0, #0, #1, FLAGS                                   ;
                                   14   AE 03190004   8F  D0 00092        MOVL    #51970052, LIST                                     ; 0605
                                   18   AE       04   AE  9E 0009A        MOVAB   OURPID, LIST+4                                      ; 0607
                                                 1C   AE  7C 0009F        CLRQ    LIST+8                                              ; 0608
                                                      7E  7C 000A2        CLRQ    -(SP)                                               ; 0611
                                                 2C   AE  9F 000A4        PUSHAB  IOSB                                                ;
                                                 20   AE  9F 000A7        PUSHAB  LIST                                                ;
                                                      7E  7C 000AA        CLRQ    -(SP)                                               ;
                                                      7E  D4 000AC        CLRL    -(SP)                                               ;
                                        68            07  FB 000AE        CALLS   #7, SYS$GETJPIW                                     ;
                                        52            50  D0 000B1        MOVL    R0, STATUS                                          ;
                                        40            52  E9 000B4        BLBC    STATUS, 1$                                          ;
                                        52       24   AE  3C 000B7        MOVZWL  IOSB, STATUS                                        ; 0612
                                        39            52  E9 000BB        BLBC    STATUS, 1$                                          ; 0613
                                                 08   AE  D4 000BE        CLRL    PID                                                 ; 0619
                                   2C   AE 020E0000   8F  D0 000C1        MOVL    #34471936, PROCNAME                                 ; 0620
                                                 30   AE  D4 000C9        CLRL    PROCNAME+4                                          ;
                                                 2C   AE  9F 000CC        PUSHAB  PROCNAME                                            ; 0621
                                                 70   A7  9F 000CF        PUSHAB  P.ADM                                               ;
                                        6A            02  FB 000D2        CALLS   #2, CLI$GET_VALUE                                   ;
                                        22            50  E9 000D5        BLBC    R0, 2$                                              ;
                                   18   AE       08   AE  9E 000D8        MOVAB   PID, LIST+4                                         ; 0624
                                                      7E  7C 000DD        CLRQ    -(SP)                                               ; 0627
                                                 2C   AE  9F 000DF        PUSHAB  IOSB                                                ;
                                                 20   AE  9F 000E2        PUSHAB  LIST                                                ;
                                                 3C   AE  9F 000E5        PUSHAB  PROCNAME                                            ;
                                                      7E  7C 000E8        CLRQ    -(SP)                                               ;
                                        68            07  FB 000EA        CALLS   #7, SYS$GETJPIW                                     ;
                                        52            50  D0 000ED        MOVL    R0, STATUS                                          ;
                                        6D            52  E9 000F0        BLBC    STATUS, 4$                                          ;
                                        52       24   AE  3C 000F3        MOVZWL  IOSB, STATUS                                        ; 0628
```

```
               66            52  E9 000F7 1$:     BLBC    STATUS, 4$                    ; 0629
                         2C  AE  9F 000FA 2$:     PUSHAB  PROCNAME                      ; 0638
                       0088  C7  9F 000FD         PUSHAB  P.ADO                         ;
               6A            02  FB 00101         CALLS   #2, CLI$GET_VALUE             ;
               5D            50  E9 00104         BLBC    R0, 5$                        ;
                         08  AE  9F 00107         PUSHAB  PID                           ; 0646
                         34  AE  DD 0010A         PUSHL   PROCNAME+4                    ; 0647
               7E        34  AE  3C 0010D         MOVZWL  PROCNAME, -(SP)               ; 0646
    00000000G  00            03  FB 00111         CALLS   #3, LIB$CVT_HTB               ;
               52            50  D0 00118         MOVL    R0, STATUS                    ;
               13            52  E8 0011B         BLBS    STATUS, 3$                    ;
                       00A0  C7  9F 0011E         PUSHAB  P.ADQ                         ; 0653
                         30  AE  9F 00122         PUSHAB  PROCNAME                      ; 0651
                             02  DD 00125         PUSHL   #2                            ;
                   0078132A  8F  DD 00127         PUSHL   #7869226                      ;
               6B            04  FB 0012D         CALLS   #4, LIB$SIGNAL                ;
                                 04 00130         RET                                   ; 0650
          14   AE  03190004  8F  D0 00131 3$:     MOVL    #51970052, LIST               ; 0662
          18   AE        08  AE  9E 00139         MOVAB   PID, LIST+4                   ; 0664
                         1C  AE  7C 0013E         CLRQ    LIST+8                        ; 0665
                             7E  7C 00141         CLRQ    -(SP)                         ; 0670
                         2C  AE  9F 00143         PUSHAB  IOSB                          ;
                         20  AE  9F 00146         PUSHAB  LIST                          ;
                             7E  D4 00149         CLRL    -(SP)                         ;
                         1C  AE  9F 0014B         PUSHAB  PID                           ;
                             7E  D4 0014E         CLRL    -(SP)                         ;
               68            07  FB 00150         CALLS   #7, SYS$GETJPIW               ;
               52            50  D0 00153         MOVL    R0, STATUS                    ;
               07            52  E9 00156         BLBC    STATUS, 4$                    ;
               52        24  AE  3C 00159         MOVZWL  IOSB, STATUS                  ; 0671
               04            52  E8 0015D         BLBS    STATUS, 5$                    ; 0673
                             52  DD 00160 4$:     PUSHL   STATUS                        ; 0676
                             3A  11 00162         BRB     9$                            ;
                         08  AE  D5 00164 5$:     TSTL    PID                           ; 0686
                             05  12 00167         BNEQ    6$                            ;
          08   AE        04  AE  D0 00169         MOVL    OURPID, PID                   ; 0687
                       00AC  C7  9F 0016E 6$:     PUSHAB  P.ADS                         ; 0694
               66            01  FB 00172         CALLS   #1, CLI$PRESENT               ;
               0F            50  E9 00175         BLBC    R0, 7$                        ;
          10   AE            0F  88 00178         BISB2   #15, FLAGS                    ; 0698
          04   AE        08  AE  D1 0017C         CMPL    PID, OURPID                   ; 0702
                             04  12 00181         BNEQ    7$                            ;
          10   AE            30  88 00183         BISB2   #48, FLAGS                    ; 0705
          08   AE        04  AE  D1 00187 7$:     CMPL    OURPID, PID                   ; 0713
                             12  13 0018C         BEQL    10$                           ;
05        10   AE            04  E0 0018E         BBS     #4, FLAGS, 8$                 ; 0714
08        10   AE            05  E1 00193         BBC     #5, FLAGS, 10$                ;
                   007812E2  8F  DD 00198 8$:     PUSHL   #7869154                      ; 0715
                             63  11 0019E 9$:     BRB     13$                           ;
                             7E  7C 001A0 10$:    CLRQ    -(SP)                         ; 0726
                         2C  AE  9F 001A2         PUSHAB  IOSB                          ;
                       01F8  C9  9F 001A5         PUSHAB  JPI_LIST                      ;
                             7E  D4 001A9         CLRL    -(SP)                         ;
                         1C  AE  9F 001AB         PUSHAB  PID                           ;
                             7E  D4 001AE         CLRL    -(SP)                         ;
               68            07  FB 001B0         CALLS   #7, SYS$GETJPIW               ;
               4B            50  E9 001B3         BLBC    STATUS, 12$                   ;
```

```
                        50      24   AE  3C 001B6        MOVZWL  IOSB, STATUS                                ; 0727
                        44           50  E9 001BA        BLBC    STATUS, 12$                                 ; 0728
                08      AE           69  D0 001BD        MOVL    FAO_PID, PID                                ; 0730
                        26      11   AE  E9 001C1        BLBC    FLAGS+1, 11$                                ; 0736
             0000'      CF           69  D0 001C5        MOVL    FAO_PID, PROC_L_PID                         ; 0739
                        50      04   A9  9E 001CA        MOVAB   FAO_PRCNAM, R0                              ; 0740
             000C       CF           50  B0 001CE        MOVW    R0, PROC_A_DESC                             ;
             0000'      CF    00AC   C9  9E 001D3        MOVAB   AUX_PRCNAM, PROC_A_DESC+4                   ; 0741
      0000'  CF   00AC  C9      04   A9  28 00 DA        MOVC3   FAO_PRCNAM, AUX_PRCNAM, PROC_Z_NAME         ; 0742
         00000000G      00           00  FB 001E3        CALLS   #0, PROC_CONT_DISPLAY                       ; 0743
                                         04 001EA        RET                                                 ; 0738
                                0C   AE  9F 001EB 11$:   PUSHAB  SCRATCH                                     ; 0754
                04      AE    8000   8F  3C 001EE        MOVZWL  #32768, 4(SP)                               ;
                                04   AE  9F 001F4        PUSHAB  4(SP)                                       ;
         00000000G      00           02  FB 001F7        CALLS   #2, LIB$GET_VM                              ;
                        06           50  E8 001FE        BLBS    STATUS, 14$                                 ;
                                     50  DD 00201 12$:   PUSHL   STATUS                                      ; 0755
                        6B           01  FB 00203 13$:   CALLS   #1, LIB$SIGNAL                              ;
                                         04 00206        RET                                                 ;
                0C      BE    8000   8F  3C 00207 14$:   MOVZWL  #32768, @SCRATCH                            ; 0756
                                04   AE  DD 0020D        PUSHL   OURPID                                      ; 0764
                                14   AE  9F 00210        PUSHAB  FLAGS                                       ;
                                14   AE  DD 00213        PUSHL   SCRATCH                                     ;
             0000V      CF           03  FB 00216        CALLS   #3, DISPLAY_DATA                            ;
                                         04 0021B        RET                                                 ; 0767

; Routine Size:  540 bytes,    Routine Base:  $CODE$ + 0000
```

```
  1 ROUTINE check_privilege : NOVALUE =
  1 !++
  1 !
  1 ! This routine checks that the image has the correct privilege.
  1 !
  1 !---
  2 BEGIN
  2
  2 LOCAL
  2     status,
  2     oldpriv : $BBLOCK[8];               ! Permanent privileges go here
  2
  2 OWN
  2     newpriv : $BBLOCK[8]                        ! Mask to disable WORLD
  2             PRESET([prv$v_world]=true);
  2
  2 !
  2 ! The image SHOW is installed with WORLD privilege, but we don't want the user
  2 ! to have that much power unless s/he already has it.  So, first check to
  2 ! see if the process has the privilege, and if not, then remove it for the
  2 ! duration of this image.
  2 !
P 3 IF NOT (status = $SETPRV(ENBFLG = 1,            ! Enable
P 3                          PRVADR = 0,            ! No new privileges
P 3                          PRMFLG = 1,            ! Permanent privs
  3                          PRVPRV = oldpriv))     ! Store current ones here
  2 THEN SIGNAL_STOP(.status);
  2 !
  2 ! Check to see if privilege there.  If not, then remove it from current
  2 ! privileges.
  2 !
  2 IF NOT .oldpriv[prv$v_world]                    ! If WORLD not permanent
  2 THEN
  3     BEGIN
P 4     IF NOT (status = $SETPRV(ENBFLG = 0,        ! Disable
P 4                              PRVADR = newpriv,  ! this privilege
P 4                              PRMFLG = 0,        ! for the duration of this image
  4                              PRVPRV = 0))
  3     THEN SIGNAL_STOP(.status)
  2     END;
  2
  2 RETURN;
  1 END;


                                        .PSECT  $OWN$,NOEXE,2

             00# 004CC NEWPRIV:.BYTE   0[2]
             01  004CE         .BYTE   1
                 004CF         .BLKB   5

                               .EXTRN  SYS$SETPRV

                               .PSECT  $CODE$,NOWRT,2

            001C 00000 CHECK_PRIVILEGE:
```

```
                                                   .WORD   Save R2,R3,R4                        ; 0768
         54 00000000G  00  9E 00002               MOVAB   SYS$SETPRV, R4                       ;
         53 00000000G  00  9E 00009               MOVAB   LIB$STOP, R3                         ;
         5E            08  C2 00010               SUBL2   #8, SP                               ;
                       5E  DD 00013               PUSHL   SP                                   ; 0793
                       01  DD 00015               PUSHL   #1                                   ;
         7E            01  7D 00017               MOVQ    #1, -(SP)                            ;
         64            04  FB 0001A               CALLS   #4, SYS$SETPRV                       ;
         52            50  D0 0001D               MOVL    R0, STATUS                           ;
         05            52  E8 00020               BLBS    STATUS, 1$                           ;
                       52  DD 00023               PUSHL   STATUS                               ; 0794
         63            01  FB 00025               CALLS   #1, LIB$STOP                         ;
         16       02   AE  E8 00028 1$:           BLBS    OLDPRIV+2, 2$                        ; 0799
                       7E  7C 0002C               CLRQ    -(SP)                                ; 0805
               0000'   CF  9F 0002E               PUSHAB  NEWPRIV                              ;
                       7E  D4 00032               CLRL    -(SP)                                ;
         64            04  FB 00034               CALLS   #4, SYS$SETPRV                       ;
         52            50  D0 00037               MOVL    R0, STATUS                           ;
         05            52  E8 0003A               BLBS    STATUS, 2$                           ;
                       52  DD 0003D               PUSHL   STATUS                               ; 0806
         63            01  FB 0003F               CALLS   #1, LIB$STOP                         ;
                           04 00042 2$:           RET                                          ; 0810

; Routine Size:  67 bytes,    Routine Base:  $CODE$ + 021C
```

```
1 ROUTINE display_data (scratch, flags, ourpid)  : NOVALUE =
2 BEGIN
2
2 !---
2 !
2 ! Display the data, based on which qualifiers the user gave.
2 !
2 ! Inputs
2 !       SCRATCH -- a handy scratch area to put stuff when we need it
2 !       FLAGS   -- the options longword to direct what to output
2 !       OURPID  -- the PID of the current (this) process
2 !
2 ! Outputs
2 !       None.  The data is displayed.
2 !
2 !---
2
2 MAP
2     scratch : REF VECTOR,
2     flags : REF $BBLOCK;
2
2 LOCAL
2     status,
2     arglst : VECTOR[3],
2     entry : REF $BBLOCK;
2
2 !
2 ! Print the header.
2 !
2 fao_systime = 0;                          ! Get current time
2 fao_terminal+4 = aux_terminal;           ! Locate the terminal string
2 fao_username+4 = aux_username;           ! Locate the username string
2 show$write_line (%ASCID '!/!%D   !16AF   User: !AF', fao_systime);
2
2 !
2 ! Display the data for each option requested.
2 !
2 IF .flags[proc$v_def]                    ! Standard info
2 THEN
3     BEGIN
3     fao_prcnam+4 = aux_prcnam;           ! Locate the process name string
3 !
3 ! If the process of interest is this process, then obtain the current
3 ! default device name and default.
3 !
3     IF .fao_pid NEQ .ourpid
3     THEN
4         BEGIN
4         fao_defdev = 0;
4         fao_defdir = cstring('Not available');
4         END
3     ELSE
4         BEGIN
4         LOCAL
4             status;
4         fao_defdev = lnm$c_namlength;            ! Length of logical disk name
4         fao_defdev + 4 = aux_defdev;             ! Where to put logical name
```

```
P 5         IF NOT (status = $TRNLOG(LOGNAM = %ASCID 'SYS$DISK',
P 5                                 RSLLEN = fao_defdev,
  5                                 RSLBUF = fao_defdev))
  4         THEN (SIGNAL(.status); RETURN);
  4         IF CH$RCHAR(aux_defdev) EQL %X'1B'
  4         THEN
  5             BEGIN
  5             fao_defdev = .fao_defdev - 4;
  5             fao_defdev + 4 = .(fao_defdev + 4) + 4;
  4             END;
  4         fao_defdir = pio$gt_ddstring;
  3         END;
  3
  3     show$write_line(%ASCID 'Pid: !XL   Proc. name: !16AF UIC: !%I',
  3                     fao_pid,
  3                     %ASCID 'Priority: !3UB   Default file spec: !AF!AC',
  3                     fao_prib);
  3 !
  3 ! Get a list of devices allocated by this process
  3 !
  3     arglst[0] = 2;
  3     arglst[1] = .scratch;
  3     arglst[2] = .fao_pid;
P 4     IF NOT (status = $CMKRNL(ROUTIN = get_devall,
  4                              ARGLST = arglst))
  3     THEN SIGNAL(.status)
  3     ELSE
  4         BEGIN
  4         entry = scratch[1];
  4         IF .entry[d_l_length] NEQ 0
  4         THEN
  5             BEGIN
  5             show$write_line(%ASCID '!/Devices allocated: !AF',
  5                             entry[d_l_length]);
  5             entry = .entry + d_k_length;
  5             WHILE .entry[d_l_length] NEQ 0 DO
  6                 BEGIN
  6                 show$write_line(%ASCID '!19< !>!AF', entry[d_l_length]);
  6                 entry = .entry + d_k_length;
  5                 END;
  4             END;
  3         END;
  3 !
  3 ! Get a list of mounted devices for this process
  3 !
  3     IF .ourpid EQL .fao_pid
  3     THEN
  4         BEGIN
  4         arglst[0] = 1;
  4         arglst[1] = .scratch;
P 5         IF NOT (status = $CMKRNL(ROUTIN = get_devmoun,
  5                                  ARGLST = arglst))
  4         THEN SIGNAL(.status)
  4         ELSE
  5             BEGIN
  5             entry = scratch[1];
```

```
;  834        0925  5               IF .entry[d_l_length] NEQ 0
;  835        0926  5               THEN
;  836        0927  6                   BEGIN
;  837        0928  6                   show$write_line(%ASCID '!/Devices mounted: !AF',
;  838        0929  6                                   entry[d_l_length]);
;  839        0930  6                   entry = .entry + d_k_length;
;  840        0931  6                   WHILE .entry[d_l_length] NEQ 0 DO
;  841        0932  7                       BEGIN
;  842        0933  7                       show$write_line(%ASCID '!17< !>!AF', entry[d_l_length]);
;  843        0934  7                       entry = .entry + d_k_length;
;  844        0935  6                       END;
;  845        0936  5                   END;
;  846        0937  4               END;
;  847        0938  3           END;
;  848        0939  2       END;
;  849        0940  2
;  850        0941  2  !
;  851        0942  2  ! Quotas
;  852        0943  2  !
;  853        0944  2  IF .flags[proc$v_quot]
;  854        0945  2  THEN
;  855        0946  3      BEGIN
;  856        0947  3  !
;  857        0948  3  ! If the CPU time limit is set to zero, then say that it's
;  858        0949  3  ! infinite.  Otherwise, mangle it and point to the mangled value.
;  859        0950  3  !
;  860        0951  3      IF .fao_cpulim EQL 0
;  861        0952  3      THEN
;  862        0953  4          BEGIN
;  863        0954  4          fao_cpulim = %CHARCOUNT('        Infinite');
;  864        0955  4          fao_cpulim + 4 = UPLIT ('        Infinite');
;  865        0956  4          END
;  866        0957  3      ELSE
;  867        0958  4          BEGIN
;  868        0959  4          LOCAL
;  869        0960  4              temp,
;  870        0961  4              quad_time : VECTOR[2];
;  871        0962  4          IF .fao_cpulim
;  872        0963  4          THEN temp = -100000
;  873        0964  4          ELSE temp = 0;
;  874        0965  4          fao_cpulim = .fao_cpulim<1,31>;
;  875        0966  4          EMUL(%REF(-200000),
;  876        0967  4               fao_cpulim,
;  877        0968  4               temp,
;  878        0969  4               quad_time);
;  879        0970  4          fao_cpulim = 16;
;  880        0971  4          fao_cpulim + 4 = aux_cpulim;
;  881      P 0972  4          $FAOL(CTRSTR = %ASCID '!%D',
;  882      P 0973  4                OUTLEN = fao_cpulim,
;  883      P 0974  4                OUTBUF = fao_cpulim,
;  884        0975  4                PRMLST = %REF(quad_time));
;  885        0976  3          END;
;  886        0977  3
;  887        0978  3  !
;  888        0979  3  ! Get the number of subprocesses remaining, by subtracting the current
;  889        0980  3  ! job process count from the process limit.
;  890        0981  3  !
```

```
3       fao_prclm = .fao_prclm - .aux_jobprccnt;
3
3       fao_account+4 = aux_account;                            ! Locate the account string
3
3       show$write_line(%ASCID '!/Process Quotas:!/ Account name: !AF',
3                       fao_account,
3                       %ASCID ' CPU limit:                 !AF  Direct I/O limit: !9UL',
3                       fao_cpulim,
3                       %ASCID ' Buffered I/O byte count quota:!10UL  Buffered I/O limit:!8UL',
3                       fao_bytcnt,
3                       %ASCID ' Timer queue entry quota:!16UL  Open file quota:!11UL',
3                       fao_tqcnt,
3                       %ASCID ' Paging file quota:!22UL  Subprocess quota:!10UL',
3                       fao_pagfilcnt,
3                       %ASCID ' Default page fault cluster:!13UL  AST limit:!17UL',
3                       fao_dfpfc,
3                       %ASCID ' Enqueue quota:!26UL  Shared file limit:!9UL',
3                       fao_enqcnt,
3                       %ASCID ' Max detached processes:!17UL  Max active jobs:!11UL',
3                       fao_maxdetach);
2       END;
2
2 !
2 ! Accounting
2 !
2 IF .flags[proc$v_acc]
2 THEN
3       BEGIN
3
3 !
3 ! Convert the CPU time to standard system format.
3 !
3       EMUL(%REF(-100000), fao_cputim, %REF(0), aux_cputim);
3       fao_cputim = aux_cputim;            ! Locate the quad cputim
3
3 !
3 ! Figure out the connect time.
3 !
4       BEGIN
4       LOCAL system_time : VECTOR[2];
4       $GETTIM(TIMADR = system_time);
4       SUBM (2, system_time, aux_logintim, aux_logintim);
3       END;
3
3       fao_logintim = aux_logintim;        ! Locate quad login time
3
3       show$write_line(%ASCID '!/Accounting information:', 0,
3                       %ASCID ' Buffered I/O count:!10UL  Peak working set size:!11UL',
3                       fao_bufio,
3                       %ASCID ' Direct I/O count:!12UL  Peak virtual size:!15UL',
3                       fao_dirio,
3                       %ASCID ' Page faults:!17UL  Mounted volumes:!17UL',
3                       fao_pageflts,
3                       %ASCID ' Images activated:!12UL',
3                       fao_imagecount,
3                       %ASCID ' Elapsed CPU time:   !%D!/ Connect time:       !%D',
3                       fao_cputim);
```

```
2       END;
2
2   !
2   ! Process privileges
2   !
2   IF .flags[proc$v_priv]
2   THEN
3       BEGIN
3       BIND privileges = aux_procpriv : BITVECTOR[64];
3       show$write_line(%ASCID ' ', 0,
3                       %ASCID 'Process privileges:',0);
3       INCR index FROM 0 TO priv_num - 1 DO
4           BEGIN
4
4           ! We do not want to display some privileges (UPGRADE, DOWNGRADE,
4           ! TMPJNL, PRMJNL) that are partially implemented but that can be set.
4           ! This looks ugly, I know, but we could not comment the privileges
4           ! out of the table because then the bits got out of synchronization.
4           !
4
4           IF .privileges[.index]
5             AND ( .index neq 32                              ! upgrade
5               AND .index neq 33                              ! downgrade
5               AND .index neq 36                              ! tmpjnl
5               AND .index neq 37 )                            ! prmjnl
4           THEN show$write_line(%ASCID ' !20AC !AC',
4                                   priv_table[.index * 2]);
3           END;
3       show$write_line(%ASCID ' ', 0);
3
3   !
3   ! Display the rights for this process.
3   !
3       IF .ourpid EQL .fao_pid
3       THEN display_rights();
2       END;
2
2
2   !
2   ! Memory
2   !
2   IF .flags[proc$v_mem]
2   THEN
3       BEGIN
4       IF NOT (status = show$prcallreg())
3       THEN SIGNAL(.status);
2       END;
2
2   !
2   ! The subprocess tree
2   !
2   IF .flags[proc$v_sub]
2   THEN display_tree(.scratch);
2
2   RETURN;
1   END;
```

```
                                                                                 .PSECT  $PLIT$,NOWRT,NOEXE,2

20  20  46  41  36  31  21  20  20  20  44  25  21  2F  21  006F0 P.ADV:  .ASCII  \!/!%D   !16AF   User: !AF\<0><0><0>
            00  00  00  46  41  21  20  3A  72  65  73  55  20  006FF
                                                  010E0019  0070C P.ADU:  .LONG   17694745
                                                  00000000' 00710         .ADDRESS P.ADV
    65  6C  62  61  6C  69  61  76  61  20  74  6F  4E  0D  00714 P.ADW:  .ASCII  <13>\Not available\
                                                            00722         .BLKB   2
                            4B  53  49  44  24  53  59  53  00724 P.ADY:  .ASCII  \SYS$DISK\
                                                  010E0008  0072C P.ADX:  .LONG   17694728
                                                  00000000' 00730         .ADDRESS P.ADY
63  6F  72  50  20  20  20  4C  58  21  20  3A  64  69  50  00734 P.AEA:  .ASCII  \Pid: !XL   Proc. name: !16AF UIC: !%I\<0>
55  20  46  41  36  31  21  20  3A  65  6D  61  6E  20  2E  00743
                            00  49  25  21  20  3A  43  49  00752
                                                    00  00  0075A         .ASCII  <0><0>
                                                  010E0025  0075C P.ADZ:  .LONG   17694757
                                                  00000000' 00760         .ADDRESS P.AEA
20  42  55  33  21  20  3A  79  74  69  72  6F  69  72  50  00764 P.AEC:  .ASCII  \Priority: !3UB   Default file spec: !AF!\
20  65  6C  69  66  20  74  6C  75  61  66  65  44  20  20  00773
                    21  46  41  21  20  3A  63  65  70  73  00782
                                            00  00  43  41  0078C         .ASCII  \AC\<0><0>
                                                  010E002A  00790 P.AEB:  .LONG   17694762
                                                  00000000' 00794         .ADDRESS P.AEC
63  6F  6C  6C  61  20  73  65  63  69  76  65  44  2F  21  00798 P.AEE:  .ASCII  \!/Devices allocated: !AF\
                        46  41  21  20  3A  64  65  74  61  007A7
                                                  010E0018  007B0 P.AED:  .LONG   17694744
                                                  00000000' 007B4         .ADDRESS P.AEE
            00  00  46  41  21  3E  21  20  3C  39  31  21  007B8 P.AEG:  .ASCII  \!19< !>!AF\<0><0>
                                                  010E000A  007C4 P.AEF:  .LONG   17694730
                                                  00000000' 007C8         .ADDRESS P.AEG
74  6E  75  6F  6D  20  73  65  63  69  76  65  44  2F  21  007CC P.AEI:  .ASCII  \!/Devices mounted: !AF\<0><0>
                        00  00  46  41  21  20  3A  64  65  007DB
                                                  010E0016  007E4 P.AEH:  .LONG   17694742
                                                  00000000' 007E8         .ADDRESS P.AEI
            00  00  46  41  21  3E  21  20  3C  37  31  21  007EC P.AEK:  .ASCII  \!17< !>!AF\<0><0>
                                                  010E000A  007F8 P.AEJ:  .LONG   17694730
                                                  00000000' 007FC         .ADDRESS P.AEK
74  69  6E  69  66  6E  49  20  20  20  20  20  20  20  20  00800 P.AEL:  .ASCII  \         Infinite\
                                                        65  0080F
                                            00  44  25  21  00810 P.AEN:  .ASCII  \!%D\<0>
                                                  010E0003  00814 P.AEM:  .LONG   17694723
                                                  00000000' 00818         .ADDRESS P.AEN
61  74  6F  75  51  20  73  73  65  63  6F  72  50  2F  21  0081C P.AEP:  .ASCII  \!/Process Quotas:!/ Account name: !AF\<0>
61  6E  20  74  6E  75  6F  63  63  41  20  2F  21  3A  73  0082B
                            00  46  41  21  20  3A  65  6D  0083A
                                                    00  00  00842         .ASCII  <0><0>
                                                  010E0025  00844 P.AEO:  .LONG   17694757
                                                  00000000' 00848         .ADDRESS P.AEP
20  20  20  20  3A  74  69  6D  69  6C  20  55  50  43  20  0084C P.AER:  .ASCII  \ CPU limit:                !AF   Direct I/O\
20  20  46  41  21  20  20  20  20  20  20  20  20  20  20  0085B
                    4F  2F  49  20  74  63  65  72  69  44  0086A
            4C  55  39  21  20  3A  74  69  6D  69  6C  20  00874         .ASCII  \ limit: !9UL\
                                                  010E0034  00880 P.AEQ:  .LONG   17694772
                                                  00000000' 00884         .ADDRESS P.AER
62  20  4F  2F  49  20  64  65  72  65  66  66  75  42  20  00888 P.AET:  .ASCII  \ Buffered I/O byte count quota:!10UL  Bu\
61   74   6F   75   71   20   74   6E   75   6F   63   20   65   74   79  00897
                         75   42   20   20   4C   55   30   31   21   3A  008A6
69   6D   69   6C   20   4F   2F   49   20   64   65   72   65   66   66  008B0           .ASCII  \ffered I/O limit:!8UL\<0><0><0>
                              00   00   00   4C   55   38   21   3A   74  008BF
                                                                010E003D  008C8 P.AES:    .LONG   17694781
                                                                00000000' 008CC           .ADDRESS P.AET
6E   65   20   65   75   65   75   71   20   72   65   6D   69   54   20  008D0 P.AEV:    .ASCII  \ Timer queue entry quota:!16UL  Open fil\
4C   55   36   31   21   3A   61   74   6F   75   71   20   79   72   74  008DF
                         6C   69   66   20   6E   65   70   4F   20   20  008EE
00   00   4C   55   31   31   21   3A   61   74   6F   75   71   20   65  008F8           .ASCII  \e quota:!11UL\<0><0><0>
                                                                      00  00907
                                                                010E0035  00908 P.AEU:    .LONG   17694773
                                                                00000000' 0090C           .ADDRESS P.AEV
75   71   20   65   6C   69   66   20   67   6E   69   67   61   50   20  00910 P.AEX:    .ASCII  \ Paging file quota:!22UL  Subprocess quo\
70   62   75   53   20   20   4C   55   32   32   21   3A   61   74   6F  0091F
                         6F   75   71   20   73   73   65   63   6F   72  0092E
                                   4C   55   30   31   21   3A   61   74  00938           .ASCII  \ta:!10UL\
                                                                010E0030  00940 P.AEW:    .LONG   17694768
                                                                00000000' 00944           .ADDRESS P.AEX
66   20   65   67   61   70   20   74   6C   75   61   66   65   44   20  00948 P.AEZ:    .ASCII  \ Default page fault cluster:!13UL  AST l\
31   21   3A   72   65   74   73   75   6C   63   20   74   6C   75   61  00957
                         6C   20   54   53   41   20   20   4C   55   33  00966
               00   00   4C   55   37   31   21   3A   74   69   6D   69  00970           .ASCII  \imit:!17UL\<0><0>
                                                                010E0032  0097C P.AEY:    .LONG   17694770
                                                                00000000' 00980           .ADDRESS P.AEZ
3A   61   74   6F   75   71   20   65   75   65   75   71   6E   45   20  00984 P.AFB:    .ASCII  \ Enqueue quota:!26UL  Shared file limit:\
66   20   64   65   72   61   68   53   20   20   4C   55   36   32   21  00993
                         3A   74   69   6D   69   6C   20   65   6C   69  009A2
                                                       4C   55   39   21  009AC           .ASCII  \!9UL\
                                                                010E002C  009B0 P.AFA:    .LONG   17694764
                                                                00000000' 009B4           .ADDRESS P.AFB
70   20   64   65   68   63   61   74   65   64   20   78   61   4D   20  009B8 P.AFD:    .ASCII  \ Max detached processes:!17UL  Max activ\
20   4C   55   37   31   21   3A   73   65   73   73   65   63   6F   72  009C7
                         76   69   74   63   61   20   78   61   4D   20  009D6
               4C   55   31   31   21   3A   73   62   6F   6A   20   65  009E0           .ASCII  \e jobs:!11UL\
                                                                010E0034  009EC P.AFC:    .LONG   17694772
                                                                00000000' 009F0           .ADDRESS P.AFD
6E   69   20   67   6E   69   74   6E   75   6F   63   63   41   2F   21  009F4 P.AFF:    .ASCII  \!/Accounting information:\<0><0><0>
          00   00   00   3A   6E   6F   69   74   61   6D   72   6F   66  00A03
                                                                010E0019  00A10 P.AFE:    .LONG   17694745
                                                                00000000' 00A14           .ADDRESS P.AFF
63   20   4F   2F   49   20   64   65   72   65   66   66   75   42   20  00A18 P.AFH:    .ASCII  \ Buffered I/O count:!10UL  Peak working \
61   65   50   20   20   4C   55   30   31   21   3A   74   6E   75   6F  00A27
                         20   67   6E   69   6B   72   6F   77   20   6B  00A36
00   4C   55   31   31   21   3A   65   7A   69   73   20   74   65   73  00A40           .ASCII  \set size:!11UL\<0><0>
                                                                      00  00A4F
                                                                010E0036  00A50 P.AFG:    .LONG   17694774
                                                                00000000' 00A54           .ADDRESS P.AFH
75   6F   63   20   4F   2F   49   20   74   63   65   72   69   44   20  00A58 P.AFJ:    .ASCII  \ Direct I/O count:!12UL  Peak virtual si\
20   6B   61   65   50   20   20   4C   55   32   31   21   3A   74   6E  00A67
                         69   73   20   6C   61   75   74   72   69   76  00A76
                                   4C   55   35   31   21   3A   65   7A  00A80           .ASCII  \ze:!15UL\
                                                                010E0030  00A88 P.AFI:    .LONG   17694768
                                                                00000000' 00A8C           .ADDRESS P.AFJ
31   21   3A   73   74   6C   75   61   66   20   65   67   61   50   20  00A90 P.AFL:    .ASCII  \ Page faults:!17UL  Mounted volumes:!17U\
6F   76   20   64   65   74   6E   75   6F   4D   20   20   4C   55   37  00A9F
                         55   37   31   21   3A   73   65   6D   75   6C  00AAE
```

```
                                    00  00  00  4C  00AB8          .ASCII   \L\<0><0><0>                                    ;
                                     010E0029  00ABC P.AFK:   .LONG    17694761                                        ;
                                     00000000'  00AC0          .ADDRESS P.AFL                                           ;
74  61  76  69  74  63  61  20  73  65  67  61  6D  49  20  00AC4 P.AFN:   .ASCII   \ Images activated:!12UL\<0>             ;
                        00  4C  55  32  31  21  3A  64  65  00AD3                                                       ;
                                     010E0017  00ADC P.AFM:   .LONG    17694743                                        ;
                                     00000000'  00AE0          .ADDRESS P.AFN                                           ;
69  74  20  55  50  43  20  64  65  73  70  61  6C  45  20  00AE4 P.AFP:   .ASCII   \ Elapsed CPU time:   !%D!/ Connect time:\ ;
6E  6F  43  20  2F  21  44  25  21  20  20  20  3A  65  6D  00AF3                                                       ;
                3A  65  6D  69  74  20  74  63  65  6E  00B02                                                           ;
        00  00  44  25  21  20  20  20  20  20  20  20  00B0C          .ASCII   \        !%D\<0><0>                     ;
                                     010E0032  00B18 P.AFO:   .LONG    17694770                                        ;
                                     00000000'  00B1C          .ADDRESS P.AFP                                           ;
                                    00  00  00  20  00B20 P.AFR:   .ASCII   \ \<0><0><0>                                ;
                                     010E0001  00B24 P.AFQ:   .LONG    17694721                                        ;
                                     00000000'  00B28          .ADDRESS P.AFR                                           ;
65  6C  69  76  69  72  70  20  73  73  65  63  6F  72  50  00B2C P.AFT:   .ASCII   \Process privileges:\<0>                  ;
                                00  3A  73  65  67  00B3B                                                               ;
                                     010E0013  00B40 P.AFS:   .LONG    17694739                                        ;
                                     00000000'  00B44          .ADDRESS P.AFT                                           ;
        00  00  43  41  21  20  43  41  30  32  21  20  00B48 P.AFV:   .ASCII   \ !20AC !AC\<0><0>                          ;
                                     010E000A  00B54 P.AFU:   .LONG    17694730                                        ;
                                     00000000'  00B58          .ADDRESS P.AFV                                           ;
                                    00  00  00  20  00B5C P.AFX:   .ASCII   \ \<0><0><0>                                ;
                                     010E0001  00B60 P.AFW:   .LONG    17694721                                        ;
                                     00000000'  00B64          .ADDRESS P.AFX                                           ;

                                                      PRIVILEGES=         AUX_LIST+356
                                                               .EXTRN   SYS$TRNLOG, SYS$CMKRNL
                                                               .EXTRN   SYS$FAOL, SYS$GETTIM

                                                               .PSECT   $CODE$,NOWRT,2

                                          01FC  00000 DISPLAY_DATA:
                                                               .WORD    Save R2,R3,R4,R5,R6,R7,R8                       ; 0811
                       58 0000000G  00  9E  00002          MOVAB    SYS$CMKRNL, R8                                  ;
                       57 0000000G  00  9E  00009          MOVAB    LIB$SIGNAL, R7                                  ;
                       56 0000000G  00  9E  00010          MOVAB    SHOW$WRITE_LINE, R6                             ;
                       55       0000'  CF  9E  00017          MOVAB    FAO_CPULIM, R5                                  ;
                       5E              18  C2  0001C          SUBL2    #24, SP                                         ;
                                C4     A5  D4  0001F          CLRL     FAO_SYSTIME                                     ; 0340
            CC     A5           64     A5  9E  00022          MOVAB    AUX_TERMINAL, FAO_TERMINAL+4                    ; 0841
            D4     A5           74     A5  9E  00027          MOVAB    AUX_USERNAME, FAO_USERNAME+4                    ; 0842
                                C4     A5  9F  0002C          PUSHAB   FAC_SYSTIME                                     ; 0843
                              0000'    CF  9F  0002F          PUSHAB   P.ADU                                           ;
                       66              02  FB  00033          CALLS    #2, SHOW$WRITE_LINE                             ;
                       54       08     AC  D0  00036          MOVL     FLAGS, R4                                       ; 0848
                       03              64  E8  0003A          BLBS     (R4), 1$                                        ;
                                    00F7  31  0003D          BRW      11$                                             ;
            E0     A5         0084     C5  9E  00040 1$:      MOVAB    AUX_PRCNAM, FAO_PRCNAM+4                        ; 0851
            0C     AC           D8     A5  D1  00046          CMPL     FAO_PID, OURPID                                 ; 0856
                                       0B  13  0004B          BEQL     2$                                              ;
                                EC     A5  D4  0004D          CLRL     FAO_DEFDEV                                      ; 0859
            F4     A5         0000'    CF  9E  00050          MOVAB    P.ADW, FAO_DEFDIR                               ; 0860
                                       40  11  00056          BRB      5$                                              ; 0856
            EC     A5           FF     8F  9A  00058 2$:      MOVZBL   #255, FAO_DEFDEV                                ; 0866
            F0     A5         0094     C5  9E  0005D          MOVAB    AUX_DEFDEV, FAO_DEFDEV+4                        ; 0867
```

```
SHOWPROCESS                                        N 11
V04-000                                  16-Sep-1984 01:25:12    VAX-11 Bliss-32 V4.0-742              Page  34
                                         14-Sep-1984 12:09:44    [CLIUTL.SRC]SHOWPROC.B32;1                 (8)

                               7E  7C 00063          CLRQ     -(SP)                               ; 0870
                               7E  D4 00065          CLRL     -(SP)                               ;
                            EC A5  9F 00067          PUSHAB   FAO_DEFDEV                          ;
                            EC A5  9F 0006A          PUSHAB   FAO_DEFDEV                          ;
                         0000' CF  9F 0006D          PUSHAB   P.ADX                               ;
00000000G        00            06  FB 00071          CALLS    #6, SYS$TRNLOG                      ;
                 06            50  E8 00078          BLBS     STATUS, 3$                          ;
                               50  DD 0007B          PUSHL    STATUS                              ; 0871
                 67            01  FB 0007D          CALLS    #1, LIB$SIGNAL                      ;
                                   04 00080          RET                                          ;
                 1B       0094 C5  91 00081 3$:      CMPB     AUX_DEFDEV, #27                     ; 0872
                               08  12 00086          BNEQ     4$                                  ;
            EC   A5            04  C2 00088          SUBL2    #4, FAO_DEFDEV                      ; 0875
            F0   A5            04  C0 0008C          ADDL2    #4, FAO_DEFDEV+4                    ; 0876
            F4   A5  00000000G 00  9E 00090 4$:      MOVAB    PIO$GT_DDSTRING, FAO_DEFDIR         ; 0878
                            E8 A5  9F 00098 5$:      PUSHAB   FAO_PRIB                            ; 0881
                         0000' CF  9F 0009B          PUSHAB   P.AEB                               ; 0882
                            D8 A5  9F 0009F          PUSHAB   FAO_PID                             ; 0881
                         0000' CF  9F 000A2          PUSHAB   P.ADZ                               ;
                 66            04  FB 000A6          CALLS    #4, SHOW$WRITE_LINE                 ;
            0C   AE            02  D0 000A9          MOVL     #2, ARGLST                          ; 0889
            10   AE         04 AC  D0 000AD          MOVL     SCRATCH, ARGLST+4                   ; 0890
            14   AE         D8 A5  D0 000B2          MOVL     FAO_PID, ARGLST+8                   ; 0891
                            0C AE  9F 000B7          PUSHAB   ARGLST                              ; 0893
                         0000V CF  9F 000BA          PUSHAB   GET_DEVALL                          ;
                 68            02  FB 000BE          CALLS    #2, SYS$CMKRNL                      ;
                 53            50  D0 000C1          MOVL     R0, STATUS                          ;
                 07            53  E8 000C4          BLBS     STATUS, 6$                          ;
                               53  DD 000C7          PUSHL    STATUS                              ; 0894
                 67            01  FB 000C9          CALLS    #1, LIB$SIGNAL                      ;
                               21  11 000CC          BRB      8$                                  ;
       52   04   AC            04  C1 000CE 6$:      ADDL3    #4, SCRATCH, ENTRY                  ; 0897
                               62  D5 000D3          TSTL     (ENTRY)                             ; 0898
                               18  13 000D5          BEQL     8$                                  ;
                               52  DD 000D7          PUSHL    ENTRY                               ; 0902
                         0000' CF  9F 000D9          PUSHAB   P.AED                               ; 0901
                 66            02  FB 000DD 7$:      CALLS    #2, SHOW$WRITE_LINE                 ; 0902
                 52            1D  C0 000E0          ADDL2    #29, ENTRY                          ; 0903
                               62  D5 000E3          TSTL     (ENTRY)                             ; 0904
                               08  13 000E5          BEQL     8$                                  ;
                               52  DD 000E7          PUSHL    ENTRY                               ; 0906
                         0000' CF  9F 000E9          PUSHAB   P.AEF                               ;
                               EE  11 000ED          BRB      7$                                  ;
            D8   A5         0C AC  D1 000EF 8$:      CMPL     OURPID, FAO_PID                     ; 0914
                               41  12 000F4          BNEQ     11$                                 ;
            0C   AE            01  D0 000F6          MOVL     #1, ARGLST                          ; 0917
            10   AE         04 AC  D0 000FA          MOVL     SCRATCH, ARGLST+4                   ; 0918
                            0C AE  9F 000FF          PUSHAB   ARGLST                              ; 0920
                         0000V CF  9F 00102          PUSHAB   GET_DEVMOUN                         ;
                 68            02  FB 00106          CALLS    #2, SYS$CMKRNL                      ;
                 53            50  D0 00109          MOVL     R0, STATUS                          ;
                 07            53  E8 0010C          BLBS     STATUS, 9$                          ;
                               53  DD 0010F          PUSHL    STATUS                              ; 0921
                 67            01  FB 00111          CALLS    #1, LIB$SIGNAL                      ;
                               21  11 00114          BRB      11$                                 ;
       52   04   AC            04  C1 00116 9$:      ADDL3    #4, SCRATCH, ENTRY                  ; 0924
                               62  D5 0011B          TSTL     (ENTRY)                             ; 0925
```

```
                                                              18  13 0011D          BEQL    11$                                         ;
                                                              52  DD 0011F          PUSHL   ENTRY                                       ; 0929
                                                   0000'      CF  9F 00121          PUSHAB  P.AEH                                       ; 0928
                                            66                02  FB 00125 10$:     CALLS   #2, SHOW$WRITE_LINE                         ; 0929
                                            52                1D  C0 00128          ADDL2   #29, ENTRY                                  ; 0930
                                                              62  D5 0012B          TSTL    (ENTRY)                                     ; 0931
                                                              08  13 0012D          BEQL    11$                                         ;
                                                              52  DD 0012F          PUSHL   ENTRY                                       ; 0933
                                                   0000'      CF  9F 00131          PUSHAB  P.AEJ                                       ;
                                                              EE  11 00135          BRB     10$                                         ;
                                 03                 64        01  E0 00137 11$:     BBS     #1, (R4), 12$                               ; 0944
                                                           0091   31 0013B          BRW     17$                                         ;
                                                    50        65  D0 0013E 12$:     MOVL    FAO_CPULIM, R0                              ; 0951
                                                              0B  12 00141          BNEQ    13$                                         ;
                                                    65        10  D0 00143          MOVL    #16, FAO_CPULIM                             ; 0954
                                           04       A5   0000'      CF  9E 00146          MOVAB   P.AEL, FAO_CPULIM+4                         ; 0955
                                                              3B  11 0014C          BRB     16$                                         ; 0951
                                                    09        50  E9 0014E 13$:     BLBC    R0, 14$                                     ; 0962
                                                    50 FFFE7960  8F  D0 00151          MOVL    #-100000, TEMP                              ; 0963
                                                              02  11 00158          BRB     15$                                         ;
                                                              50  D4 0015A 14$:     CLRL    TEMP                                        ; 0964
           65                    65                 1F        01  EF 0015C 15$:     EXTZV   #1, #31, FAO_CPULIM, FAO_CPULIM             ; 0965
    04     AE                    50                 65 FFFCF2C0  8F  7A 00161          EMUL    #-200000, FAO_CPULIM, TEMP, QUAD_TIME       ; 0966
                                                    65        10  D0 0016B          MOVL    #16, FAO_CPULIM                             ; 0970
                                           04       A5     01A4  C5  9E 0016E          MOVAB   AUX_CPULIM, FAO_CPULIM+4                    ; 0971
                                                    6E       04  AE  9E 00174          MOVAB   QUAD_TIME, (SP)                             ; 0975
                                                           4020  8F  BB 00178          PUSHR   #^M<R5,SP>                                  ;
                                                              55  DD 0017C          PUSHL   R5                                          ;
                                                   0000'      CF  9F 0017E          PUSHAB  P.AEM                                       ;
                                    00000000G       00        04  FB 00182          CALLS   #4, SYS$FAOL                                ;
                                           20       A5     01B4  C5  C2 00189 16$:     SUBL2   AUX_JOBPRCCNT, FAO_PRCLM                    ; 0982
                                           FC       A5     0194  C5  9E 0018F          MOVAB   AUX_ACCOUNT, FAO_ACCOUNT+4                  ; 0984
                                                             34  A5  9F 00195          PUSHAB  FAO_MAXDETACH                               ; 0986
                                                          0000'  CF  9F 00198          PUSHAB  P.AFC                                       ; 0999
                                                             2C  A5  9F 0019C          PUSHAB  FAO_ENQCNT                                  ; 0986
                                                          0000'  CF  9F 0019F          PUSHAB  P.AFA                                       ; 0997
                                                             24  A5  9F 001A3          PUSHAB  FAO_DFPFC                                   ; 0986
                                                          0000'  CF  9F 001A6          PUSHAB  P.AEY                                       ; 0995
                                                             1C  A5  9F 001AA          PUSHAB  FAO_PAGFILCNT                               ; 0986
                                                          0000'  CF  9F 001AD          PUSHAB  P.AEW                                       ; 0993
                                                             14  A5  9F 001B1          PUSHAB  FAO_TQCNT                                   ; 0986
                                                          0000'  CF  9F 001B4          PUSHAB  P.AEU                                       ; 0991
                                                             0C  A5  9F 001B8          PUSHAB  FAO_BYTCNT                                  ; 0986
                                                          0000'  CF  9F 001BB          PUSHAB  P.AES                                       ; 0989
                                                                 55  DD 001BF          PUSHL   R5                                          ; 0986
                                                          0000'  CF  9F 001C1          PUSHAB  P.AEQ                                       ; 0987
                                                             F8  A5  9F 001C5          PUSHAB  FAO_ACCOUNT                                 ; 0986
                                                          0000'  CF  9F 001C8          PUSHAB  P.AEO                                       ;
                                                    66           10  FB 001CC          CALLS   #16, SHOW$WRITE_LINE                        ;
                                 5A                 64           02  E1 001CF 17$:     BBC     #2, (R4), 18$                               ; 1007
   01B8     C5                   00        58       A5 FFFE7960  8F  7A 001D3          EMUL    #-100000, FAO_CPUTIM, #0, AUX_CPUTIM        ; 1014
                                           58       A5     01B8  C5  9E 001DF          MOVAB   AUX_CPUTIM, FAO_CPUTIM                      ; 1015
                                                             04  AE  9F 001E5          PUSHAB  SYSTEM_TIME                                 ; 1022
                                    00000000G       00           01  FB 001E8          CALLS   #1, SYS$GETTIM                              ;
                                         01C0       C5       04  AE  C2 001EF          SUBL2   SYSTEM_TIME, AUX_LOGINTIM                   ; 1023
                                         01C4       C5       08  AE  D9 001F5          SBWC    SYSTEM_TIME, AUX_LOGINTIM+4                 ;
                                           5C       A5     01C0  C5  9E 001FB          MOVAB   AUX_LOGINTIM, FAO_LOGINTIM                  ; 1026
```

```
                               58   A5  9F 00201          PUSHAB  FAO_CPUTIM                               ; 1028
                            0000'   CF  9F 00204          PUSHAB  P.AFO                                    ; 1036
                               54   A5  9F 00208          PUSHAB  FAO_IMAGECOUNT                           ; 1028
                            0000'   CF  9F 0020B          PUSHAB  P.AFM                                    ; 1034
                               4C   A5  9F 0020F          PUSHAB  FAO_PAGEFLTS                             ; 1028
                            0000'   CF  9F 00212          PUSHAB  P.AFK                                    ; 1032
                               44   A5  9F 00216          PUSHAB  FAO_DIRIO                                ; 1028
                            0000'   CF  9F 00219          PUSHAB  P.AFI                                    ; 1030
                               3C   A5  9F 0021D          PUSHAB  FAO_BUFIO                                ; 1028
                            0000'   CF  9F 00220          PUSHAB  P.AFG                                    ;
                                    7E  D4 00224          CLRL    -(SP)                                    ;
                            0000'   CF  9F 00226          PUSHAB  P.AFE                                    ;
                    66              0C  FB 0022A          CALLS   #12, SHOW$WRITE_LINE                     ;
  54                64              03  E1 0022D 18$:     BBC     #3, (R4), 21$                            ; 1044
                                    7E  D4 00231          CLRL    -(SP)                                    ; 1048
                            0000'   CF  9F 00233          PUSHAB  P.AFS                                    ;
                                    7E  D4 00237          CLRL    -(SP)                                    ;
                            0000'   CF  9F 00239          PUSHAB  P.AFQ                                    ;
                    66              04  FB 0023D          CALLS   #4, SHOW$WRITE_LINE                      ;
                                    52  D4 00240          CLRL    INDEX                                    ; 1050
  24        01C8    C5              52  E1 00242 19$:     BBC     INDEX, PRIVILEGES, 20$                   ; 1059
                    20              52  D1 00248          CMPL    INDEX, #32                               ; 1060
                                    1F  13 0024B          BEQL    20$                                      ;
                    21              52  D1 0024D          CMPL    INDEX, #33                               ; 1061
                                    1A  13 00250          BEQL    20$                                      ;
                    24              52  D1 00252          CMPL    INDEX, #36                               ; 1062
                                    15  13 00255          BEQL    20$                                      ;
                    25              52  D1 00257          CMPL    INDEX, #37                               ; 1063
                                    10  13 0025A          BEQL    20$                                      ;
  50                52              01  78 0025C          ASHL    #1, INDEX, R0                            ; 1065
                        0354 C540       DF 00260          PUSHAL  PRIV_TABLE[R0]                           ;
                            0000'   CF  9F 00265          PUSHAB  P.AFU                                    ; 1064
                    66              02  FB 00269          CALLS   #2, SHOW$WRITE_LINE                      ; 1065
  D2                52              26  F3 0026C 20$:     AOBLEQ  #38, INDEX, 19$                          ; 1050
                                    7E  D4 00270          CLRL    -(SP)                                    ; 1067
                            0000'   CF  9F 00272          PUSHAB  P.AFW                                    ;
                    66              02  FB 00276          CALLS   #2, SHOW$WRITE_LINE                      ;
              D8    A5         0C   AC  D1 00279          CMPL    OURPID, FAO_PID                          ; 1072
                                    05  12 0027E          BNEQ    21$                                      ;
           0000V    CF              00  FB 00280          CALLS   #0, DISPLAY_RIGHTS                       ; 1073
  12                64              04  E1 00285 21$:     BBC     #4, (R4), 22$                            ; 1080
       00000000G    00              00  FB 00289          CALLS   #0, SHOW$PRCALLREG                       ; 1083
                    53              50  D0 00290          MOVL    R0, STATUS                               ;
                    05              53  E8 00293          BLBS    STATUS, 22$                              ;
                                    53  DD 00296          PUSHL   STATUS                                   ; 1084
                    67              01  FB 00298          CALLS   #1, LIB$SIGNAL                           ;
  08                64              05  E1 0029B 22$:     BBC     #5, (R4), 23$                            ; 1090
                               04   AC  DD 0029F          PUSHL   SCRATCH                                  ; 1091
           0000V    CF              01  FB 002A2          CALLS   #1, DISPLAY_TREE                         ;
                                    04  002A7 23$:        RET                                              ; 1094

; Routine Size:  680 bytes,    Routine Base:  $CODE$ + 025F
```

```
  1 ROUTINE display_tree (scratch) : NOVALUE =
  2 BEGIN
  2
  2 !---
  2 !
  2 ! A routine to print the subprocess tree.  This is used as the starting
  2 ! point for calling NEXT_PROCESS, which is a recursive routine.  This
  2 ! routine simply sets up the master process, the process at the top of
  2 ! the tree, and then calls NEXT_PROCESS to wind its way down the tree.
  2 ! When control is finally returned to this routine, then all the processes
  2 ! in the tree have been displayed.
  2 !
  2 ! Inputs
  2 !       SCRATCH -- address of the scratch area that contains all the
  2 !                  info on the subprocesses
  2 !
  2 ! Outputs
  2 !       None.  The subprocess tree is output.
  2 !
  2 !---
  2
  2 MAP
  2     scratch : REF VECTOR;                   ! The first longword contains the
  2                                             ! size of the scratch area
  2 LOCAL
  2     fao_pix,                                ! Index for the fao pid
  2     master_pix : WORD,                      ! Master index for this JIB set
  2     entry : REF $BBLOCK,                    ! Pointer to the entries
  2     arglst : VECTOR[3],                     ! The ever-present argument list
  2     status;                                 ! sigh...
  2
  2 !
  2 ! Get information on all the relevant processes.
  2 !
  2 arglst[0] = 1;                              ! Set up the argument list
  2 arglst[1] = .scratch;
P 3 IF NOT (status = $CMKRNL(ROUTIN = make_tree,
  3                          ARGLST = arglst))
  2 THEN (SIGNAL(.status); RETURN);
  2
  2 !
  2 ! Find the master process.
  2 !
  2 entry = scratch[1];                         ! Set up a pointer to the entries
  2
  2 WHILE .entry[sub_owner] NEQ 0 DO            ! Look for a non-owned process
  2     entry = entry[sub_name] + pcb$s_lname;
  2
  2 master_pix = .entry[sub_pix];               ! Got it, now save it away.
  2
  2 !
  2 ! Display this master process.  If it is the originator, say so.
  2 !
  2 show$write_line(%ASCID '!/Processes in this tree: !/', 0);
  2
  2 arglst[0] = entry[sub_name];                ! Point to the process name.
  2 fao_pix = .fao_pid<0,.sch$gl_pixwidth>;     ! Get index for fao process
```

```
; 1062          1152  2 IF .master_pix EQL .fao_pix             ! If this is the originator
; 1063          1153  2 THEN arglst[1] = cstring(' (*)')        ! then point it out
; 1064          1154  2 ELSE arglst[1] = UPLIT(0);              ! otherwise be silent
; 1065          1155  2 show$write_line(%ASCID '!AC!AC', arglst);
; 1066          1156  2
; 1067          1157  2 !
; 1068          1158  2 ! Now go get all the other subprocesses hanging off this one.
; 1069          1159  2 !
; 1070          1160  2 entry = scratch[1];                      ! Always start at beginning of list
; 1071          1161  2 next_process(.entry, .master_pix, .fao_pix, 1);
; 1072          1162  2
; 1073          1163  2 RETURN;
; 1074          1164  1 END;


                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

20  6E  69  20  73  65  73  73  65  63  6F  72  50  2F  21  00B68 P.AFZ:  .ASCII  \!/Processes in this tree: !/\       ;
        2F  21  20  3A  65  65  72  74  20  73  69  68  74  00B77                                                       ;
                                          010E001C  00B84 P.AFY:  .LONG   17694748                              ;
                                          00000000' 00B88         .ADDRESS P.AFZ                                ;
                              29  2A  28  20  04  00B8C P.AGA:  .ASCII  <4>\ (*)\                             ;
                                                  00B91           .BLKB   3
                                          00000000  00B94 P.AGB:  .LONG   0                                     ;
              00  00  43  41  21  43  41  21  00B98 P.AGD:  .ASCII  \!AC!AC\<0><0>                        ;
                                          010E0006  00BA0 P.AGC:  .LONG   17694726                              ;
                                          00000000' 00BA4         .ADDRESS P.AGD                                ;


                                                                        .PSECT  $CODE$,NOWRT,2

                                              007C 00000 DISPLAY_TREE:
                                                                  .WORD   Save R2,R3,R4,R5,R6                   ; 1095
                    56 00000000G   00   9E 00002         MOVAB   SHOW$WRITE_LINE, R6                   ;
                    5E             08   C2 00009         SUBL2   #8, SP                                ;
                                   01   DD 0000C         PUSHL   #1                                    ; 1129
              04    AE       04    AC   D0 0000E         MOVL    SCRATCH, ARGLST+4                     ; 1130
                                   5E   DD 00013         PUSHL   SP                                    ; 1132
                           0000V   CF   9F 00015         PUSHAB  MAKE_TREE                             ;
              00000000G    00      02   FB 00019         CALLS   #2, SYS$CMKRNL                        ;
                           0A      50   E8 00020         BLBS    STATUS, 1$                            ;
                                   50   DD 00023         PUSHL   STATUS                                ; 1133
              00000000G    00      01   FB 00025         CALLS   #1, LIB$SIGNAL                        ;
                                        04 0002C         RET                                           ;
         55         04     AC      04   C1 0002D 1$:     ADDL3   #4, SCRATCH, R5                       ; 1138
                           52      55   D0 00032         MOVL    R5, ENTRY                             ;
                                   62   B5 00035 2$:     TSTW    (ENTRY)                               ; 1140
                                   05   13 00037         BEQL    3$                                    ;
                           52      14   C0 00039         ADDL2   #20, ENTRY                            ; 1141
                                   F7   11 0003C         BRB     2$                                    ;
                           54   02 A2   B0 0003E 3$:     MOVW    2(ENTRY), MASTER_PIX                  ; 1143
                                   7E   D4 00042         CLRL    -(SP)                                 ; 1148
                           0000'   CF   9F 00044         PUSHAB  P.AFY                                 ;
                           66      02   FB 00048         CALLS   #2, SHOW$WRITE_LINE                   ;
                           6E   04 A2   9E 0004B         MOVAB   4(R2), ARGLST                         ; 1150
```

```
      53     0000'  CF 00000000G  00           00  EF 0004F          EXTZV   #0, SCH$GL_PIXWIDTH, FAO_PID, FAO_PIX     ; 1151
      53            54            10           00  ED 0005A          CMPZV   #0, #16, MASTER_PIX, FAO_PIX             ; 1152
                                               08  12 0005F          BNEQ    4$                                       ;
                          04      AE   0000'   CF  9E 00061          MOVAB   P.AGA, ARGLST+4                          ; 1153
                                               06  11 00067          BRB     5$                                       ;
                          04      AE   0000'   CF  9E 00069 4$:      MOVAB   P.AGB, ARGLST+4                          ; 1154
                                               5E  DD 0006F 5$:      PUSHL   SP                                       ; 1155
                                       0000'   CF  9F 00071          PUSHAB  P.AGC                                    ;
                                  66           02  FB 00075          CALLS   #2, SHOW$WRITE_LINE                      ;
                                  52           55  D0 00078          MOVL    R5, ENTRY                                ; 1160
                                               01  DD 0007B          PUSHL   #1                                       ; 1161
                                               53  DD 0007D          PUSHL   FAO_PIX                                  ;
                                  7E           54  3C 0007F          MOVZWL  MASTER_PIX, -(SP)                        ;
                                               52  DD 00082          PUSHL   ENTRY                                    ;
                          0000V   CF           04  FB 00084          CALLS   #4, NEXT_PROCESS                         ;
                                               04  00089             RET                                              ; 1164

; Routine Size:  138 bytes,    Routine Base:  $CODE$ + 0507
```

```
1 ROUTINE next_process (entry, pid, origin, level) : NOVALUE =
2 BEGIN
2
2 !---
2 !
2 ! A magical little routine that looks for a process in the scratch
2 ! area that has an owner PID equal to the PID that is passed to it.
2 ! If such a process is found, then this subroutine is called again,
2 ! with that process's PID, to get all its children.  It's a fairly
2 ! nifty way to go about printing the tree.
2 !
2 ! Inputs
2 !       ENTRY  -- beginning address of scratch area
2 !       PID    -- the PID to look for as a process's owner
2 !       ORIGIN -- the PID of the process that this SHOW PROCESS
2 !                 request was issued for.
2 !       LEVEL  -- a handy way to tell what level of subprocess we're at.
2 !
2 ! Outputs
2 !       The subprocess tree is displayed.
2 !
2 !---
2
2 LOCAL
2     status,
2     list : REF $BBLOCK;
2
2 list = .entry;                                  ! Point to the list of entries
2
2 !
2 ! Scan the entries, looking for an entry whose owner field is equal to PID.
2 ! If/when such an entry is found, print it.  If that entry is the process
2 ! of interest, indicate it.
2 !
2 DO
3     BEGIN
3     IF .list[sub_owner] EQL .pid<0,16>          ! If this process is owned by
3     THEN                                        ! the process, print it.
4         BEGIN
4         LOCAL
4             temp : VECTOR[80,BYTE],             ! Need a place to store spaces
4             arglst : VECTOR[3];                 ! And an argument list
4         temp[0] = 2 * .level;                   ! Print this many leading
4         CH$FILL(' ', 2*.level+1, temp[1]);      ! spaces before the name.
4         arglst[0] = temp;                       ! Now point to it.
4         arglst[1] = list[sub_name];             ! Point to the process name
4         IF .list[sub_pix] EQL .origin<0,16>     ! If the originator,
4         THEN arglst[2] = cstring(' (*)')        ! then point it out
4         ELSE arglst[2] = UPLIT(0);              ! otherwise be silent
4         show$write_line(%ASCID '!AC!AC!AC', arglst);
4 !
4 ! Zero the owner field, so that this process isn't displayed again.  Then
4 ! call this subroutine with this process's PID, and a higher level, to display
4 ! all of its children.
4 !
4         list[sub_owner] = 0;
4         next_process(.entry, .list[sub_pix], .origin, .level+1);
```

```
; 1133        1222  3         END;
; 1134        1223  3     list = [list[sub_name] + pcb$s_lname;
; 1135        1224  3     END
; 1136        1225  2 UNTIL .list[sub_pix] EQL 0;
; 1137        1226  2
; 1138        1227  2 RETURN;
; 1139        1228  1 END;


                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                   29  2A  28  20  04  00BA8 P.AGE:     .ASCII  <4>\ (*)\                                 ;
                                                       00BAD            .BLKB   3
                                          00000000     00BB0 P.AGF:     .LONG   0                                         ;
      00  00  00  43  41  21  43  41  21  43  41  21   00BB4 P.AGH:     .ASCII  \!AC!AC!AC\<0><0><0>                      ;
                                          010E0009     00BC0 P.AGG:     .LONG   17694729                                  ;
                                          00000000'    00BC4            .ADDRESS P.AGH                                    ;


                                                                        .PSECT  $CODE$,NOWRT,2

                                               007C 00000 NEXT_PROCESS:
                                                                        .WORD   Save R2,R3,R4,R5,R6                       ; 1165
                             5E          A4  AE  9E 00002               MOVAB   -92(SP), SP                               ;
                             56          04  AC  D0 00006               MOVL    ENTRY, LIST                               ; 1192
                        08   AC              66  B1 0000A 1$:           CMPW    (LIST), PID                               ; 1201
                                             52  12 0000E               BNEQ    4$                                        ;
                 50     10   AC              01  78 00010               ASHL    #1, LEVEL, R0                             ; 1207
                        0C   AE              50  90 00015               MOVB    R0, TEMP                                  ;
                                             50  D6 00019               INCL    R0                                        ; 1208
      50         20          6E              00  2C 0001B               MOVC5   #0, (SP), #32, R0, TEMP+1                 ;
                                         0D  AE     00020                                                                 ;
                             6E          0C  AE  9E 00022               MOVAB   TEMP, ARGLST                              ; 1209
                        04   AE          04  A6  9E 00026               MOVAB   4(R6), ARGLST+4                           ; 1210
                        0C   AC          02  A6  B1 0002B               CMPW    2(LIST), ORIGIN                           ; 1211
                                             08  12 00030               BNEQ    2$                                        ;
                        08   AE       0000'  CF  9E 00032               MOVAB   P.AGE, ARGLST+8                           ; 1212
                                             06  11 00038               BRB     3$                                        ;
                        08   AE       0000'  CF  9E 0003A 2$:           MOVAB   P.AGF, ARGLST+8                           ; 1213
                                             5E  DD 00040 3$:           PUSHL   SP                                        ; 1214
                                      0000'  CF  9F 00042               PUSHAB  P.AGG                                     ;
                   00000000G  00             02  FB 00046               CALLS   #2, SHOW$WRITE_LINE                       ;
                                             66  B4 0004D               CLRW    (LIST)                                    ; 1220
                 7E     10   AC              01  C1 0004F               ADDL3   #1, LEVEL, -(SP)                          ; 1221
                                         0C  AC  DD 00054               PUSHL   ORIGIN                                    ;
                             7E          02  A6  3C 00057               MOVZWL  2(LIST), -(SP)                            ;
                                         04  AC  DD 0005B               PUSHL   ENTRY                                     ;
                        9E   AF              04  FB 0005E               CALLS   #4, NEXT_PROCESS                          ;
                             56              14  C0 00062 4$:           ADDL2   #20, LIST                                 ; 1223
                                         02  A6  B5 00065               TSTW    2(LIST)                                   ; 1225
                                             A0  12 00068               BNEQ    1$                                        ;
                                                 04 0006A               RET                                               ; 1228

; Routine Size:  107 bytes,    Routine Base:  $CODE$ + 0591
```

```
ROUTINE make_tree (scratch) =
BEGIN

!---
!
! This routine operates in KERNEL mode.  It locates the JIB, which is shared
! by all processes in this tree.  Then, it scans all the processes in the
! system, gathering information about each process with the appropriate JIB
! address.  All the data is stored in the scratch area pointed to by SCRATCH.
!
! Inputs
!       SCRATCH -- address of the scratch area.
!
! Outputs
!       SCRATCH -- will contain all relevant data about the subprocesses.
!
!---

MAP
    scratch : REF VECTOR;

LOCAL
    entry : REF $BBLOCK,
    pcb : REF $BBLOCK,
    limit,
    jib_address;

entry = scratch[1];                             ! Point to the scratch area
limit = scratch[0] + .scratch[0] -256;          ! Limit is half a page before
                                                ! end of scratch area.
!
! Get the JIB address.  All processes of interest will have the same JIB address
!
pcb = exe$epid_to_pcb (.fao_pid);
jib_address = .pcb[pcb$l_jib];

!
! Scan all the processes and get only those whose JIB address matches
! this process's JIB address.
!
INCR index FROM sch$c_swppix+1 TO .sch$gl_maxpix DO
    BEGIN
    pcb = .sch$gl_pcbvec[.index];
    IF .pcb[pcb$l_jib] EQL .jib_address
    AND .entry LSSA .limit
    THEN
        BEGIN
        entry[sub_pix] = .pcb[pcb$l_pid];
        entry[sub_owner] = .pcb[pcb$l_owner];
        entry = CH$MOVE(pcb$s_lname, pcb[pcb$t_lname], entry[sub_name]);
        END;
    END;

entry[sub_pix] = entry[sub_owner] = 0;  ! To show end of list

RETURN 1;
END;
```

```
                                                 03FC 00000 MAKE_TREE:
                                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                   : 1229
               53      04   AC                04  C1 00002         ADDL3   #4, SCRATCH, ENTRY                             : 1256
               56      04   AC         04     BC  C1 00007         ADDL3   @SCRATCH, SCRATCH, R6                          : 1257
                            56       FF00     C6  9E 0000D         MOVAB   -256(R6), LIMIT                                :
                            50       0000'    CF  D0 00012         MOVL    FAO_PID, R0                                    : 1262
                               00000000G      00  16 00017         JSB     EXE$EPID_TO_PCB                                :
                            57                50  D0 0001D         MOVL    R0, PCB                                        :
                            59       0080     C7  D0 00020         MOVL    128(PCB), JIB_ADDRESS                          : 1263
               58 00000000G 8F                01  C3 00025         SUBL3   #1, #SCH$C_SWPPIX+1, INDEX                     : 1272
                                              26  11 0002D         BRB     2$                                             :
                            50 00000000G      00  D0 0002F 1$:     MOVL    SCH$GL_PCBVEC, R0                              : 1271
                            57              6048  D0 00036         MOVL    (R0)[INDEX], PCB                               :
                            59       0080     C7  D1 0003A         CMPL    128(PCB), JIB_ADDRESS                          : 1272
                                              14  12 0003F         BNEQ    2$                                             :
                            56                53  D1 00041         CMPL    ENTRY, LIMIT                                   : 1273
                                              0F  1E 00044         BGEQU   2$                                             :
                       02   A3         60     A7  B0 00046         MOVW    96(PCB), 2(ENTRY)                              : 1276
                            63         1C     A7  B0 0004B         MOVW    28(PCB), (ENTRY)                               : 1277
          04   A3      70   A7                10  28 0004F         MOVC3   #16, 112(PCB), 4(ENTRY)                        : 1278
               D2           58 00000000G      00  F3 00055 2$:     AOBLEQ  SCH$GL_MAXPIX, INDEX, 1$                       : 1269
                                              63  D4 0005D         CLRL    (ENTRY)                                        : 1282
                            50                01  D0 0005F         MOVL    #1, R0                                         : 1284
                                                  04 00062         RET                                                    : 1285

; Routine Size:  99 bytes,    Routine Base:  $CODE$ + 05FC
```

```
1 ROUTINE get_devall (data, pid) =
2 BEGIN
2
2 !---
2 !
2 ! This routine operates in KERNEL mode, and gathers the names of all devices
2 ! allocated by this process.
2 !
2 ! Inputs
2 !       DATA -- scratch area to store the names
2 !       PID  -- PID of the particular process of interest
2 !
2 ! Outputs
2 !       DATA -- will contain the device names
2 !
2 !---
2
2 MAP
2     data : REF VECTOR;
2
2 LOCAL
2     status,
2     ipid,                                       ! Internal pid
2     limit,                                      ! End-of-address limit
2     scratch : REF $BBLOCK,                      ! Pointer to scratch area
2     ucb : REF $BBLOCK,                          ! UCB pointer
2     ddb : REF $BBLOCK;                          ! DDB pointer
2
2 !
2 ! Set up the scratch area so that is can be addressed easily.  Also, calculate
2 ! a limit toward the end of the scratch area, so that we don't write beyond the
2 ! area.  Finally, set up STATUS as 1, to show that we still have room in the
2 ! scratch area to store more data.
2 !
2 scratch = data[1];                              ! Point to beginning of scratch area
2 limit = .data[0] + data[0] - 256;               ! Set the limit to be halfway in to
2                                                 ! the last page of the scratch area.
2 status = 1;                                     ! Indicate no problem yet.
2
2 !
2 ! Lock the I/O data base.  Upon return from the call to SCH$IOLOCKR, the
2 ! IPL will be 2, so that pagefaults are still allowed.
2 !
2 SCH$IOLOCKR(.ctl$gl_pcb);                       ! Lock the I/O database
2
2 !
2 ! Convert the extended PID to an internal PID for checking the I/O database.
2 !
2 ipid = exe$epid_to_ipid (.pid);
2
2 !
2 ! For each UCB in the I/O database, see if the owner PID matches the
2 ! internal PID of interest
2 !
2 status = IOC$SCAN_IODB(0, 0; ddb, ucb);
2 WHILE .status DO
3     BEGIN
```

```
; 1257          1343  3      IF .ucb[ucb$l_pid] EQL .ipid
; 1258          1344  3      THEN
; 1259          1345  4          BEGIN
; 1260          1346  4          IF .scratch GEQA .limit          ! Check if there is still room
; 1261          1347  4          THEN (status = SS$_VASFULL; EXITLOOP);
; 1262          1348  4
; 1263          1349  4          IF ioc$cvt_devnam(21,                      ! Get device name, max this long
; 1264          1350  4                            scratch[d_t_device],     ! put it here,
; 1265          1351  4                            -1,                      ! in standard display format
; 1266          1352  4                            .ucb;                    ! UCB is here
; 1267          1353  4                            scratch[d_l_length])     ! final length here
; 1268          1354  4          THEN
; 1269          1355  5              BEGIN
; 1270          1356  5              scratch[d_l_length] = .scratch[d_l_length] - 1;
; 1271          1357  5              scratch[d_a_ptr] = scratch[d_t_device] + 1;
; 1272          1358  5              scratch = .scratch + d_k_length;
; 1273          1359  4              END;
; 1274          1360  3          END;
; 1275          1361  3      status = IOC$SCAN_IODB(.ddb, .ucb; ddb, ucb);
; 1276          1362  2      END;
; 1277          1363  2
; 1278          1364  2  scratch[d_l_length] = 0;                           ! To show end of list
; 1279          1365  2  IF .status EQL 0                                   ! If just end of list
; 1280          1366  2  THEN status = 1;                                   ! then readjust status
; 1281          1367  2
; 1282          1368  2  !
; 1283          1369  2  ! Now to clean up.  Unlock the I/O database, then lower the IPL
; 1284          1370  2  ! to zero.
; 1285          1371  2  !
; 1286          1372  2  SCH$IOUNLOCK(.ctl$gl_pcb);                         ! Unlock I/O database
; 1287          1373  2  SET_IPL(0);                                        ! Lower IPL
; 1288          1374  2
; 1289          1375  2  RETURN .status;                                    ! Return with status
; 1290          1376  1  END;                                               ! End of GET_DATA



                                         0FFC 00000 GET_DEVALL:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1286
                              58 00000000G  00  9E 00002        MOVAB   CTL$GL_PCB, R8                              ;
                  53      04  AC            04  C1 00009        ADDL3   #4, DATA, SCRATCH                           ; 1320
                  52      04  BC        04  AC  C1 0000E        ADDL3   DATA, @DATA, R2                             ; 1321
                              52      FF00  C2  9E 00014        MOVAB   -256(R2), LIMIT                             ;
                              56            01  D0 00019        MOVL    #1, STATUS                                  ; 1323
                              54            68  D0 0001C        MOVL    CTL$GL_PCB, R4                              ; 1329
                                 00000000G  00  16 0001F        JSB     SCH$IOLOCKR                                 ;
                              50        08  AC  D0 00025        MOVL    PID, R0                                     ; 1334
                                 00000000G  00  16 00029        JSB     EXE$EPID_TO_IPID                            ;
                              57            50  D0 0002F        MOVL    R0, IPID                                    ;
                                            5A  7C 00032        CLRQ    R10                                         ; 1340
                                 00000000G  00  16 00034 1$:    JSB     IOC$SCAN_IODB                               ;
                              56            50  D0 0003A        MOVL    R0, STATUS                                  ;
                              37            56  E9 0003D        BLBC    STATUS, 3$                                  ; 1341
                              57        2C  AA  D1 00040        CMPL    44(UCB), IPID                               ; 1343
                                            EE  12 00044        BNEQ    1$                                          ;
```

```
               52             53  D1 00046        CMPL    SCRATCH, LIMIT                              ; 1346
                              07  1F 00049        BLSSU   2$                                          ;
               56       0244  8F  3C 0004B        MOVZWL  #580, STATUS                                ; 1347
                              25  11 00050        BRB     3$                                          ;
               51         08  A3  9E 00052 2$:    MOVAB   8(SCRATCH), R1                              ; 1350
               55             5A  D0 00056        MOVL    UCB, R5                                     ; 1353
               54             01  CE 00059        MNEGL   #1, R4                                      ;
               50             15  D0 0005C        MOVL    #21, R0                                     ;
                  00000000G   00  16 0005F        JSB     IOC$CVT_DEVNAM                              ;
               63             51  D0 00065        MOVL    R1, (SCRATCH)                               ;
               C9             50  E9 00068        BLBC    R0, 1$                                      ;
                              63  D7 0006B        DECL    (SCRATCH)                                   ; 1356
          04   A3         09  A3  9E 0006D        MOVAB   9(R3), 4(SCRATCH)                           ; 1357
               53             1D  C0 00072        ADDL2   #29, SCRATCH                                ; 1358
                              BD  11 00075        BRB     1$                                          ; 1361
                              63  D4 00077 3$:    CLRL    (SCRATCH)                                   ; 1364
                              56  D5 00079        TSTL    STATUS                                      ; 1365
                              03  12 0007B        BNEQ    4$                                          ;
               56             01  D0 0007D        MOVL    #1, STATUS                                  ; 1366
               54             68  D0 00080 4$:    MOVL    CTL$GL_PCB, R4                              ; 1372
                  00000000G   00  16 00083        JSB     SCH$IOUNLOCK                                ;
               12             00  DA 00089        MTPR    #0, #18                                     ; 1373
               50             56  D0 0008C        MOVL    STATUS, R0                                  ; 1375
                                  04 0008F        RET                                                 ; 1376

; Routine Size:  144 bytes,    Routine Base:  $CODE$ + 065F
```

```
1 ROUTINE get_devmoun (data) =
2 BEGIN
2
2 !---
2 !
2 ! This one operates in KRNL mode, and simply goes down the mounted volume
2 ! list for this process, putting the name of the device(s) into the DATA
2 ! area.
2 !
2 ! Inputs
2 !       DATA -- scratch area to store the device names
2 !
2 ! Outputs
2 !       DATA -- will contain some device names, maybe.
2 !
2 !---
2
2 MAP data : REF VECTOR;
2
2 LOCAL
2     status,
2     limit,
2     mtl_head,
2     jib     : REF $BBLOCK,
2     devlist : REF $BBLOCK,
2     scratch : REF $BBLOCK;
2
2 BIND
2     pcb = .ctl$gl_pcb : $BBLOCK;
2
2 !
2 ! Set up the scratch area so that is can be addressed easily.  Also, calculate
2 ! a limit toward the end of the scratch area, so that we don't write beyond the
2 ! area.  Finally, set up STATUS as 1, to show that we still have room in the
2 ! scratch area to store more data.
2 !
2 scratch = data[1];                                  ! Point to beginning of scratch area
2 limit = .data[0] + data[0] - 256;                   ! Set the limit to be halfway in to
2                                                     ! the last page of the scratch area.
2 status = 1;                                         ! Indicate no problem yet.
2
2 !
2 ! Starting at the mounted device list in the JIB. Simply copy
2 ! the device name and unit number into the scratch area.
2 !
2 jib = .pcb [pcb$l_jib];                             ! get the JIB address
2 mtl_head = devlist = jib [jib$l_mtlfl];             ! get the mount list head
2 sch$iolockr(.ctl$gl_pcb);                           ! lock I/O database
2
2
2 WHILE .status                                                ! While no error
2 AND (devlist = .devlist[mtl$l_mtlfl]) NEQ .mtl_head DO  ! not at end of
3     BEGIN                                                    ! the list, copy
3     IF .scratch GEQA .limit                         ! Check if there is still room
3     THEN (status = SS$_VASFULL; EXITLOOP);
3     IF ioc$cvt_devnam(21,                           ! Get device name, max this long
3                       scratch[d_t_device],          ! put it here,
; 1349          1434  3                     -1,                         ! in standard display format
; 1350          1435  3                     .devlist[mtl$l_ucb];        ! UCB is here
; 1351          1436  3                     scratch[d_l_length])        ! final length here
; 1352          1437  3         THEN
; 1353          1438  4             BEGIN
; 1354          1439  4             scratch[d_l_length] = .scratch[d_l_length] -1;
; 1355          1440  4             scratch[d_a_ptr] = scratch[d_t_device] + 1;
; 1356          1441  4             scratch = .scratch + d_k_length;
; 1357          1442  3             END;
; 1358          1443  2         END;
; 1359          1444  2
; 1360          1445  2     sch$iounlock(.ctl$gl_pcb);                          ! Unlock I/O database
; 1361          1446  2     set_ipl(0);                                         ! and drop IPL
; 1362          1447  2     scratch[d_l_length] = 0;                            ! Zero to show end of list
; 1363          1448  2
; 1364          1449  2     RETURN .status;
; 1365          1450  1     END;



                                      0FFC 00000 GET_DEVMOUN:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1377
                59 00000000'      00  9E 00002          MOVAB   CTL$GL_PCB, R9                                ;
                54                59  D0 00009          MOVL    CTL$GL_PCB, R4                                ; 1405
    53      04  AC                04  C1 0000C          ADDL3   #4, DATA, SCRATCH                             ; 1413
    52      04  BC          04    AC  C1 00011          ADDL3   DATA, @DATA, R2                               ; 1414
                52        FF00    C2  9E 00017          MOVAB   -256(R2), LIMIT                               ;
                57                01  D0 0001C          MOVL    #1, STATUS                                    ; 1416
                50        0080    C4  D0 0001F          MOVL    128(R4), JIB                                  ; 1422
                56                50  D0 00024          MOVL    JIB, DEVLIST                                  ; 1423
                58                50  D0 00027          MOVL    JIB, MTL_HEAD                                 ;
                   00000000G      00  16 0002A          JSB     SCH$IOLOCKR                                   ; 1424
                3A                57  E9 00030 1$:      BLBC    STATUS, 3$                                    ; 1427
                56                66  D0 00033          MOVL    (DEVLIST), DEVLIST                            ; 1428
                58                56  D1 00036          CMPL    DEVLIST, MTL_HEAD                             ;
                                  32  13 00039          BEQL    3$                                            ;
                52                53  D1 0003B          CMPL    SCRATCH, LIMIT                                ; 1430
                                  07  1F 0003E          BLSSU   2$                                            ;
                57        0244    8F  3C 00040          MOVZWL  #580, STATUS                                  ; 1431
                                  26  11 00045          BRB     3$                                            ;
                51          08    A3  9E 00047 2$:      MOVAB   8(SCRATCH), R1                                ; 1433
                55          0C    A6  D0 0004B          MOVL    12(DEVLIST), R5                               ; 1436
                54                01  CE 0004F          MNEGL   #1, R4                                        ;
                50                15  D0 00052          MOVL    #21, R0                                       ;
                   00000000G      00  16 00055          JSB     IOC$CVT_DEVNAM                                ;
                63                51  D0 0005B          MOVL    R1, (SCRATCH)                                 ;
                CF                50  E9 0005E          BLBC    R0, 1$                                        ;
                                  63  D7 00061          DECL    (SCRATCH)                                     ; 1439
            04  A3          09    A3  9E 00063          MOVAB   9(R3), 4(SCRATCH)                             ; 1440
                53                1D  C0 00068          ADDL2   #29, SCRATCH                                  ; 1441
                                  C3  11 0006B          BRB     1$                                            ; 1427
                54                69  D0 0006D 3$:      MOVL    CTL$GL_PCB, R4                                ; 1445
                   00000000G      00  16 00070          JSB     SCH$IOUNLOCK                                  ;
                12                00  DA 00076          MTPR    #0, #18                                       ; 1446
                                  63  D4 00079          CLRL    (SCRATCH)                                     ; 1447
```

```
                                        50          57  D0 0007B          MOVL    STATUS, R0                              ; 1449
                                                        04 0007E          RET                                             ; 1450

; Routine Size:  127 bytes,    Routine Base:  $CODE$ + 06EF
```

```
ROUTINE display_rights : NOVALUE =
BEGIN

!+++
!
! Display the identifiers found in the process rights lists.
!
! Inputs:
!       None.
!
! Outputs:
!       None.  The process rights are displayed.
!
!---

LOCAL
    status,
    size,
    id_block : REF VECTOR,
    name_buffer : VECTOR[30],
    name_desc : VECTOR[2];

!
! Fi'st, get the total number of ID's that are in the rights list.
!
IF NOT (status = $CMKRNL(ROUTIN = get_rights_size,
                         ARGLST = size))
THEN (SIGNAL(.status); RETURN);

!
! Now grab a chunk of memory large enough to put the ID's.
!
IF NOT (status = lib$get_vm(%REF(8*.size), id_block))
THEN (SIGNAL(.status); RETURN);


!
! Get the ID's
!
IF NOT (status = $CMKRNL(ROUTIN = get_rights,
                         ARGLST = .id_block))
THEN (SIGNAL(.status); RETURN);

!
! If the second ID block is zero, then there is nothing to display.
!
IF .id_block[2] EQL 0
THEN RETURN;

!
! Print a header.
!
show$write_line(%ASCID 'Process rights identifiers:', 0);

name_desc[1] = name_buffer;

!
```

```
; 1424          1508  2 ! Run thru the ID's, skipping the UIC identifier.
; 1425          1509  2 !
; 1426          1510  2 INCR i FROM 1 TO .size-1 DO
; 1427          1511  3     BEGIN
; 1428          1512  3     IF .id_block[2*.i] NEQ 0
; 1429          1513  3     THEN
; 1430          1514  4         BEGIN
; 1431          1515  4         name_desc[0] = %ALLOCATION(name_buffer);
; 1432        P 1516  5         IF NOT (status = $IDTOASC(ID      = .id_block[2*.i],
; 1433        P 1517  5                                   NAMLEN = name_desc,
; 1434          1518  5                                   NAMBUF = name_desc))
; 1435          1519  4         THEN SIGNAL(.status)
; 1436          1520  4         ELSE show$write_line(%ASCID ' !AS', %REF(name_desc));
; 1437          1521  3         END;
; 1438          1522  2     END;
; 1439          1523  2
; 1440          1524  2 RETURN;
; 1441          1525  1 END;


                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

20  73  74  68  67  69  72  20  73  73  65  63  6F  72  50  00BC8 P.AGJ:  .ASCII  \Process rights identifiers:\<0>        ;
        0C  3A  73  72  65  69  66  69  74  6E  65  64  69  00BD7                                                             ;
                                          010E001B  00BE4 P.AGI:  .LONG   17694747                              ;
                                          00000C00' 00BE8         .ADDRESS P.AGJ                                ;
                                    53  41  21  20  00BEC P.AGL:  .ASCII  \ !AS\                                ;
                                          010E0004  00BF0 P.AGK:  .LONG   17694724                              ;
                                          00000000' 00BF4         .ADDRESS P.AGL                                ;

                                                                  .EXTRN  SYS$IDTOASC

                                                                  .PSECT  $CODE$,NOWRT,2

                                             00FC 00000 DISPLAY_RIGHTS:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7             ; 1451
                       57 00000000G  00  9E 00002                 MOVAB   SHOW$WRITE_LINE, R7                ;
                       56 00000000G  00  9E 00009                 MOVAB   LIB$SIGNAL, R6                     ;
                       55 00000000G  00  9E 00010                 MOVAB   SYS$CMKRNL, R5                     ;
                       5E       FF74  CE  9E 00017                MOVAB   -140(SP), SP                       ;
                                  04  AE  9F 0001C                PUSHAB  SIZE                               ; 1477
                               0000V  CF  9F 0001F                PUSHAB  GET_RIGHTS_SIZE                    ;
                       65             02  FB 00023                CALLS   #2, SYS$CMKRNL                     ;
                       54             50  D0 00026                MOVL    R0, STATUS                         ;
                       2C             54  E9 00029                BLBC    STATUS, 1$                         ;
                                  08  AE  9F 0002C                PUSHAB  ID_BLOCK                           ; 1483
         04   AE      08   AE             03  78 0002F            ASHL    #3, SIZE, 4(SP)                    ;
                                  04  AE  9F 00035                PUSHAB  4(SP)                              ;
                00000000G  00             02  FB 00038            CALLS   #2, LIB$GET_VM                     ;
                           54             50  D0 0003F            MOVL    R0, STATUS                         ;
                           13             54  E9 00042            BLBC    STATUS, 1$                         ;
                           53         08  AE  D0 00045            MOVL    ID_BLOCK, R3                       ; 1491
                                          53  DD 00049            PUSHL   R3                                 ;
                               0000V  CF  9F 0004B                PUSHAB  GET_RIGHTS                         ;
                           65             02  FB 0004F            CALLS   #2, SYS$CMKRNL                     ;
                           54             50  D0 00052            MOVL    R0, STATUS                         ;
```

```
                  06              54  E8 00055       BLBS    STATUS, 2$                              ;
                                  54  DD 00058 1$:   PUSHL   STATUS                                  ; 1492
                  66              01  FB 0005A       CALLS   #1, LIB$SIGNAL                          ;
                                      04 0005D       RET                                             ;
                            08    A3  D5 0005E 2$:   TSTL    8(R3)                                   ; 1497
                                  53  13 00061       BEQL    6$                                      ;
                                  7E  D4 00063       CLRL    -(SP)                                   ; 1503
                         0000'    CF  9F 00065       PUSHAB  P.AGI                                   ;
                  67              02  FB 00069       CALLS   #2, SHOW$WRITE_LINE                     ;
             10   AE        14    AE  9E 0006C       MOVAB   NAME_BUFFER, NAME_DESC+4                ; 1505
                                  52  D4 00071       CLRL    I                                       ; 1510
                                  3C  11 00073       BRB     5$                                      ;
       50         52              01  78 00075 3$:   ASHL    #1, I, R0                               ; 1512
                                6340  D5 00079       TSTL    (R3)[R0]                                ;
                                  33  13 0007C       BEQL    5$                                      ;
             0C   AE        78    8F  9A 0007E       MOVZBL  #120, NAME_DESC                         ; 1515
                                  7E  7C 00083       CLRQ    -(SP)                                   ; 1518
                                  7E  D4 00085       CLRL    -(SP)                                   ;
                            18    AE  9F 00087       PUSHAB  NAME_DESC                               ;
                            1C    AE  9F 0008A       PUSHAB  NAME_DESC                               ;
                                6340  DD 0008D       PUSHL   (R3)[R0]                                ;
       00000000G  00              06  FB 00090       CALLS   #6, SYS$IDTOASC                         ;
                  54              50  D0 00097       MOVL    R0, STATUS                              ;
                  07              54  E8 0009A       BLBS    STATUS, 4$                              ;
                                  54  DD 0009D       PUSHL   STATUS                                  ; 1519
                  66              01  FB 0009F       CALLS   #1, LIB$SIGNAL                          ;
                                  0D  11 000A2       BRB     5$                                      ;
                  6E        0C    AE  9E 000A4 4$:   MOVAB   NAME_DESC, (SP)                         ; 1520
                                  5E  DD 000A8       PUSHL   SP                                      ;
                         0000'    CF  9F 000AA       PUSHAB  P.AGK                                   ;
                  67              02  FB 000AE       CALLS   #2, SHOW$WRITE_LINE                     ;
       BF         52        04    AE  F2 000B1 5$:   AOBLSS  SIZE, I, 3$                             ; 1510
                                      04 000B6 6$:   RET                                             ; 1525

; Routine Size:  183 bytes,    Routine Base:  $CODE$ + 076E
; 1443          1526  1 ROUTINE get_rights_size =
; 1444          1527  2 BEGIN
; 1445          1528  2
; 1446          1529  2 !++++
; 1447          1530  2 !
; 1448          1531  2 ! Calculate the size of the rights list
; 1449          1532  2 !
; 1450          1533  2 !       THIS ROUTINE OPERATES IN KERNEL MODE
; 1451          1534  2 !
; 1452          1535  2 ! Inputs:
; 1453          1536  2 !       AP = address of size longword
; 1454          1537  2 !
; 1455          1538  2 ! Outputs:
; 1456          1539  2 !       AP gets filled with the number of rights ID's found.
; 1457          1540  2 !
; 1458          1541  2 !----
; 1459          1542  2
; 1460          1543  2 BUILTIN
; 1461          1544  2     ap;
; 1462          1545  2
; 1463          1546  2 BIND
; 1464          1547  2     pcb = .ctl$gl_pcb : $BBLOCK,
; 1465          1548  2     arb = .pcb[pcb$l_arb] : $BBLOCK,
; 1466          1549  2     rightslist = arb[arb$l_rightslist] : VECTOR;
; 1467          1550  2
; 1468          1551  2 REGISTER
; 1469          1552  2     size;
; 1470          1553  2
; 1471          1554  2 size = 0;
; 1472          1555  2
; 1473          1556  2 !
; 1474          1557  2 ! For each rights list described in the rightslist vector, add
; 1475          1558  2 ! the size of the rights list.
; 1476          1559  2 !
; 1477          1560  2 INCR i FROM 0 TO 3 DO
; 1478          1561  3     BEGIN
; 1479          1562  3     BIND
; 1480          1563  3         rights_desc = .rightslist[.i] : $BBLOCK;
; 1481          1564  3     IF .rightslist[.i] NEQ 0
; 1482          1565  3     THEN size = .size + .rights_desc[dsc$w_length];
; 1483          1566  2     END;
; 1484          1567  2
; 1485          1568  2 !
; 1486          1569  2 ! The size is in bytes, and each rights ID is 8 bytes.  So, return
; 1487          1570  2 ! the number of ID's, NOT the size in bytes.
; 1488          1571  2 !
; 1489          1572  2 .ap = .size/8;
; 1490          1573  2
; 1491          1574  2 RETURN 1;
; 1492          1575  1 END;



                                                  001C 00000 GET_RIGHTS_SIZE:
                                                                 .WORD   Save R2,R3,R4                          ; 1526
```

```
                             50 00000000G  00   D0 00002           MOVL    CTL$GL_PCB, R0                   ; 1547
            53      008C     C0            20   C1 00009           ADDL3   #32, 140(R0), R3                 ; 1549
                                           50   7C 0000F           CLRQ    SIZE                             ; 1554
                             52          6341   D0 00011 1$:       MOVL    (R3)[I], R2                      ; 1563
                                           06   13 00015           BEQL    2$                               ; 1564
                             54            62   3C 00017           MOVZWL  (R2), R4                         ; 1565
                             50            54   C0 0001A           ADDL2   R4, SIZE                         ;
            F0               51            03   F3 0001D 2$:       AOBLEQ  #3, I, 1$                        ; 1560
            6C               50            08   C7 00021           DIVL3   #8, SIZE, (AP)                   ; 1572
                             50            01   D0 00025           MOVL    #1, R0                           ; 1574
                                                04 00028           RET                                      ; 1575

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 0825
; 1494          1576  1 ROUTINE get_rights =
; 1495          1577  2 BEGIN
; 1496          1578  2
; 1497          1579  2 !++++
; 1498          1580  2 !
; 1499          1581  2 ! This routine copies the local rights into a user-readable area.
; 1500          1582  2 ! This routine executes in KERNEL mode.
; 1501          1583  2 !
; 1502          1584  2 ! Inputs:
; 1503          1585  2 !       AP = address of local user area
; 1504          1586  2 !
; 1505          1587  2 ! Outputs:
; 1506          1588  2 !       None.  The data is copied to the user area.
; 1507          1589  2 !
; 1508          1590  2 !---
; 1509          1591  2
; 1510          1592  2 BUILTIN
; 1511          1593  2     ap;
; 1512          1594  2
; 1513          1595  2 BIND
; 1514          1596  2     pcb = .ctl$gl_pcb : $BBLOCK,
; 1515          1597  2     arb = .pcb[pcb$l_arb] : $BBLOCK,
; 1516          1598  2     rightslist = arb[arb$l_rightslist] : VECTOR;
; 1517          1599  2
; 1518          1600  2 LOCAL
; 1519          1601  2     ptr;
; 1520          1602  2
; 1521          1603  2 ptr = .ap;
; 1522          1604  2
; 1523          1605  2 !
; 1524          1606  2 ! For each rights list described in the PCB rightslist vector,
; 1525          1607  2 ! if the size is non-zero, copy the contents of the rights list
; 1526          1608  2 ! to the user area.
; 1527          1609  2 !
; 1528          1610  2 INCR i FROM 0 TO 3 DO
; 1529          1611  3     BEGIN
; 1530          1612  3     BIND
; 1531          1613  3         rights_desc = .rightslist[.i] : $BBLOCK;
; 1532          1614  3     IF .rightslist[.i] NEQ 0
; 1533          1615  3     THEN IF .rights_desc[dsc$w_length] NEQ 0
; 1534          1616  3     THEN ptr = CH$MOVE(.rights_desc[dsc$w_length],
; 1535          1617  3                       .rights_desc[dsc$a_pointer],
; 1536          1618  3                       .ptr);
; 1537          1619  2     END;
; 1538          1620  2
; 1539          1621  2 RETURN 1;
; 1540          1622  1 END;



                                       00FC 00000 GET_RIGHTS:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7                   ; 1576
                       50 00000000G  00  D0 00002        MOVL    CTL$GL_PCB, R0                           ; 1596
     57      008C      C0            20  C1 00009        ADDL3   #32, 140(R0), R7                         ; 1598
                       53            5C  D0 0000F        MOVL    AP, PTR                                  ; 1603
```

```
                                       56  D4 00012         CLRL    I                                                   ; 1610
                        50           6746  D0 00014 1$:     MOVL    (R7)[I], R0                                         ; 1613
                                       09  13 00018         BEQL    2$                                                  ; 1614
                                       60  B5 0001A         TSTW    (R0)                                                ; 1615
                                       05  13 0001C         BEQL    2$                                                  ;
           63          04       B0     60  28 0001E         MOVC3   (R0), @4(R0), (PTR)                                 ; 1618
           ED                   56     03  F3 00023 2$:     AOBLEQ  #3, I, 1$                                           ; 1610
                                50     01  D0 00027         MOVL    #1, R0                                              ; 1621
                                           04 0002A         RET                                                         ; 1622

; Routine Size:  43 bytes,    Routine Base:  $CODE$ + 084E
; 1542          1623  1 END
; 1543          1624  0 ELUDOM


                                                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                                       PSECT SUMMARY
;
;       Name                       Bytes                          Attributes
;
;    $OWN$                            1236  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $PLIT$                           3064  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $GLOBAL$                           28  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $CODE$                           2169  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]LIB.L32;1              18619       76         0       1000           00:01.9




; Information:  1
; Warnings:     0
; Errors:       0



;                                       COMMAND QUALIFIERS

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

; Size:          2169 code + 4328 data bytes
; Run Time:         00:51.4
; Elapsed Time:     02:51.6
; Lines/CPU Min:     1896
; Lexemes/CPU-Min: 24098
; Memory Used:  294 pages
; Compilation Complete
```

0057 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY