I 3

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

```
                                                   K 10
**FILE**ID**SETMISC




  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  MM      MM    IIIIII      SSSSSSSS    CCCCCCCC
  SSSSSSSS  EEEEEEEEEE  TTTTTTTTTT  MM      MM    IIIIII      SSSSSSSS    CCCCCCCC
SS          EE              TT      MMMM  MMMM      II      SS          CC
SS          EE              TT      MMMM  MMMM      II      SS          CC
SS          EE              TT      MM  MM  MM      II      SS          CC
SS          EE              TT      MM  MM  MM      II      SS          CC
  SSSSSS    EEEEEEEE        TT      MM      MM      II        SSSSSS    CC
  SSSSSS    EEEEEEEE        TT      MM      MM      II        SSSSSS    CC
        SS  EE              TT      MM      MM      II              SS  CC
        SS  EE              TT      MM      MM      II              SS  CC
        SS  EE              TT      MM      MM      II              SS  CC             ....
        SS  EE              TT      MM      MM      II              SS  CC             ....
SSSSSSSS    EEEEEEEEEE      TT      MM      MM    IIIIII    SSSSSSSS      CCCCCCCC     ....
SSSSSSSS    EEEEEEEEEE      TT      MM      MM    IIIIII    SSSSSSSS      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 setmisc ( IDENT = 'V04-000',
0                 ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL=LONG_RELATIVE)
0                 ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 ! FACILITY:  SETPRO Command
1 !
1 ! ABSTRACT:
1 !
1 !       This module sets various parameters in the system.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX/VMS operating system. Privileged user mode.
1 !
1 ! AUTHOR:  Gerry Smith                          12-Jan-1983
1 !
1 ! Modified by:
1 !
1 !       V03-010 AEW0001         Anne E. Warner          24-Jul-1984
1 !               Add a check to see if the qualifier is present before
1 !               getting the value to the following qualifiers:
1 !                   /INTERACTIVE in SET$LOGINS
1 !                   /BLOCK_COUNT in SET$RMS_DEFAULT
1 !                   /BUFFER_COUNT in SET$RMS_DEFAULT
1 !                   /PROLOGUE in SET$RMS_DEFAULT
1 !                   /EXTEND_QUANTITY in SET$RMS_DEFAULT
1 !                   /NETWORK_BLOCK_COUNT in SET$RMS_DEFAULT
1 !               This check is insure correct behavior with negated qualifiers
1 !
1 !       V03-009 DAS0001         David Solomon           09-Jul-1984
1 !               Fix truncation errors; make nonexternal refs LONG_RELATIVE.
```

```
;   1 !
;   1 !      V03-008 RAS0281          Ron Schaefer          27-Mar-1984
;   1 !              Add Network Block Count to SET/RMS command.
;   1 !
;   1 !      V03-007 MCN0155          Maria del C. Nasr     01-Mar-1984
;   1 !              The disallow flag offset in the PCB is from the beginning
;   1 !              of the structure, and not a status flag.  This will fix
;   1 !              the behavior of the /ADJUST qualifier.
;   1 !
;   1 !      V03-006 GAS0172                    Gerry Smith     25-Aug-1983
;   1 !              When enabling logins, use a symbolic, UCB$V_TT_NOLOGINS,
;   1 !              instead of dead-reckoning.
;   1 !
;   1 !      V03-005 GAS0158                    Gerry Smith     25-Jul-1983
;   1 !              For SET LOGIN/INTER=0, do not disable the creation of
;   1 !              interactive jobs.
;   1 !
;   1 !      V03-004 GAS0134                    Gerry Smith     17-May-1983
;   1 !              For SET WORKING_SET, use twice the number of fluid pages,
;   1 !              rather than one.
;   1 !
;   1 !      V03-003 GAS0112                    Gerry Smith     29-Mar-1983
;   1 !              Remove all references to the old CLI interface.
;   1 !
;   1 !      V03-002 GAS0111                    Gerry Smith      9-Mar-1983
;   1 !              Fix the output of SET LOGIN.  Also calculate a better
;   1 !              minimum working set to use as a limit in SET WORKING_SET.
;   1 !
;   1 !      V03-001 GAS0110                    Gerry Smith     28-Feb-1983
;   1 !              Fix a couple of bugs with SET RMS and SET WORKING_SET,
;   1 !              caused by incorrectly computing the new RMS limit, and
;   1 !              the new working set parameters.
;   1 !
;   1 !--
```

```
  1 !
  1 ! Include files
  1 !
  1 LIBRARY 'SYS$LIBRARY:LIB';                   ! VAX/VMS common definitions
  1
  1 !
  1 ! Define the bit offsets for the SET DAY qualifier flags byte.
  1 !
  1 MACRO
  1     set$v_primary   = 0, 2, 1, 0%,
  1     set$v_secondary = 0, 3, 1, 0%,
  1     set$v_default   = 0, 4, 1, 0%;
  1
  1 !
  1 ! Define the bits for the SET RMS command
  1 !
  1 MACRO
  1     set$v_system = 0, 2, 1, 0%,                    ! /SYSTEM
  1     set$v_block  = 0, 3, 1, 0%,                    ! Block count specified
  1     set$v_buffer = 0, 4, 1, 0%,                    ! Buffer count specified
  1     set$v_prolog = 0, 5, 1, 0%,                    ! Prologue level specified
  1     set$v_disk   = 0, 6, 1, 0%,                    ! /DISK
  1     set$v_tape   = 0, 7, 1, 0%,                    ! /MAGTAPE
  1     set$v_unit   = 0, 8, 1, 0%,                    ! /UNIT_RECORD
  1     set$v_seq    = 0, 9, 1, 0%,                    ! /SEQUENTIAL
  1     set$v_rel    = 0,10, 1, 0%,                    ! /RELATIVE
  1     set$v_index  = 0,11, 1, 0%,                    ! /INDEXED
  1     set$v_hash   = 0,12, 1, 0%,                    ! /HASHED (maybe someday)
  1     set$v_extend = 0,13, 1, 0%,                    ! /EXTEND_QUANTITY
  1     set$v_netblk = 0,14, 1, 0%;                    ! /NETWORK Block Count
  1
  1 !
  1 ! Define some bits for the SET WORKING_SET command
  1 !
  1 MACRO
  1     set$v_log    = 0, 0, 1, 0%,                    ! /[NO]LOG
  1     set$v_explog = 0, 1, 1, 0%,                    ! /[NO]LOG set explicitly
  1     set$v_limit  = 0, 2, 1, 0%,                    ! /LIMIT
  1     set$v_quota  = 0, 3, 1, 0%,                    ! /QUOTA
  1     set$v_extent = 0, 4, 1, 0%,                    ! /EXTENT
  1     set$v_expadj = 0, 5, 1, 0%,                    ! /[NO]ADJUST set explicitly
  1     set$v_adjust = 0, 6, 1, 0%;                    ! /[NO]ADJUST
  1
  1 !
  1 ! Declare some shared messages
  1 !
P 1 $SHR_MSGDEF      (SET,119,LOCAL,
P 1                  (confqual,       error),
P 1                  (invquaval,      error),
  1                  (valerr,         error));
  1
```

```
1 !
1 ! Table of contents
1 !
1
1 FORWARD ROUTINE
1     set$day : NOVALUE,                        ! Set the day primary or secondary
1     setdayknl,                                ! Kernel mode routine to set the day
1     set$login : NOVALUE,                      ! Set the number of interactive users
1     setlogknl,                                ! Kernel mode routine to set logins
1     set$rms_default : NOVALUE,                ! Set the various RMS defaults
1     setrmsknl,                                ! Kernel mode routine to set RMS
1     set$working_set : NOVALUE,                ! Set the working set parameters
1     setwrkknl;                                ! Kernel mode routine to set working set
1
1 !
1 ! External routines
1 !
1 EXTERNAL ROUTINE
1     lib$cvt_dtb,                              ! Convert ASCII to binary
1     cli$get_value,                            ! Get value from CLI
1     cli$present;                              ! See if qualifier is present
1
1 !
1 ! External references
1 !
1 EXTERNAL
1     exe$gl_flags : $BBLOCK,                   ! The general system flagword
1     ctl$gl_pcb,                       ! Address of this process's PCB
1     ctl$gl_phd,                               ! Process-mapped PHD
1     ctl$gq_procpriv : $BBLOCK,                ! Process privilege mask
1     sys$gl_jobctlmb : $BBLOCK,                ! Job controller mailbox
1     sys$gw_ijobcnt : WORD,                    ! Number of current interactive jobs
1     sys$gw_ijoblim : WORD,                    ! Interactive job limit
1                                               ! Multiblock counts
1     sys$gb_dfmbc : BYTE,                      ! (system)
1     pio$gb_dfmbc : BYTE,                      ! (process)
1     sys$gb_dfnbc : BYTE,                      ! (system) Network
1     pio$gb_dfnbc : BYTE,                      ! (process)
1                                               ! Prologue levels
1     sys$gb_rmsprolog : BYTE,                  ! (system)
1     pio$gb_rmsprolog : BYTE,                  ! (process)
1                                               ! Default extend quantities
1     sys$gw_rmsextend : WORD,                  ! (system)
1     pio$gw_rmsextend : WORD,                  ! (process)
1                                               ! Multibuffer counts
1     sys$gb_dfmbfsdk : BYTE,                   ! Disk (system)
1     sys$gb_dfmbfsmt : BYTE,                   ! Tape (system)
1     sys$gb_dfmbfsur : BYTE,                   ! Unit_record (system)
1     sys$gb_dfmbfidx : BYTE,                   ! Indexed files (system)
1     sys$gb_dfmbfhsh : BYTE,                   ! Hashed files (system)
1     sys$gb_dfmbfrel : BYTE,                   ! Relative files (system)
1     pio$gb_dfmbfsdk : BYTE,                   ! Disk (process)
1     pio$gb_dfmbfsmt : BYTE,                   ! Tape (process)
1     pio$gb_dfmbfsur : BYTE,                   ! Unit_record (process)
1     pio$gb_dfmbfidx : BYTE,                   ! Indexed files (process)
1     pio$gb_dfmbfhsh : BYTE,                   ! Hashed files (process)
1     pio$gb_dfmbfrel : BYTE;                   ! Relative files (process)
```

```
;  202          0200   1
;  203          0201   1 !
;  204          0202   1 ! Declare literals defined elsewhere
;  205          0203   1 !
;  206          0204   1 EXTERNAL LITERAL
;  207          0205   1     exe$v_explicitp,                     ! Flags to show whether the day is
;  208          0206   1     exe$v_explicits,                     ! secondary or primary
;  209          0207   1     cli$_absent,                         ! CLI flag saying qualifier absent
;  210          0208   1     set$_newlims,                        ! Informational message for SET WORKING_SET
;  211          0209   1     set$_intset;                         ! Informational message for SET LOGIN
;  212          0210   1
;  214          0211  1 GLOBAL ROUTINE set$day : NOVALUE =
;  215          0212  2 BEGIN
;  216          0213  2 !++
;  217          0214  2 ! Functional description
;  218          0215  2 !
;  219          0216  2 !       This is the routine for the SET DAY command.  It is called from the
;  220          0217  2 !       SET command processor,  and sets the day to be either primary or
;  221          0218  2 !       secondary, or sets it back to its default.
;  222          0219  2 !
;  223          0220  2 ! Inputs
;  224          0221  2 !       None
;  225          0222  2 !
;  226          0223  2 ! Outputs
;  227          0224  2 !       None
;  228          0225  2 !
;  229          0226  2 !----
;  230          0227  2
;  231          0228  2 LOCAL
;  232          0229  2     status,                             ! Status return
;  233          0230  2     arglst : VECTOR[2],                 ! Argument list for $CMKRNL
;  234          0231  2     flags : $BBLOCK[1]                  ! Flags byte,
;  235          0232  2         INITIAL(BYTE(0));               ! originally zero
;  236          0233  2
;  237          0234  2 !
;  238          0235  2 ! Find out what the day is supposed to be set to.
;  239          0236  2 !
;  240          0237  2 flags[set$v_secondary] = cli$present(%ASCID 'SECONDARY');
;  241          0238  2 flags[set$v_primary]   = cli$present(%ASCID 'PRIMARY');
;  242          0239  2 flags[set$v_default]   = cli$present(%ASCID 'DEFAULT');
;  243          0240  2
;  244          0241  2 !
;  245          0242  2 ! See if the user has the OPER privilege.  If not, signal an error.
;  246          0243  2 !
;  247          0244  2 IF NOT .ctl$gq_procpriv[prv$v_oper]            ! User must have OPER priv.
;  248          0245  2 THEN SIGNAL_STOP(ss$_nooper);
;  249          0246  2
;  250          0247  2 !
;  251          0248  2 ! Change mode to kernel and set the day.
;  252          0249  2 !
;  253          0250  2 arglst[0] = 1;
;  254          0251  2 arglst[1] = flags;
;  255        P 0252  3 IF NOT (status = $CMKRNL(ROUTIN = setdayknl,
;  256          0253  3                            ARGLST = arglst))
;  257          0254  2 THEN SIGNAL_STOP(.status);
;  258          0255  2
;  259          0256  2 RETURN 1;
;  260          0257  1 END;


                                                                   .TITLE  SETMISC
                                                                   .IDENT  \V04-000\

                                                                   .PSECT  $PLIT$,NOWRT,NOEXE,2

         00  00  00  59  52  41  44  4E  4F  43  45  53  00000 P.AAB:  .ASCII  \SECONDARY\<0><0><0>              ;
                                           010E0009  0000C P.AAA:  .LONG   17694729                           ;
                                           00000000' 00010         .ADDRESS P.AAB                             ;
```

```
                     00  59  52  41  4D  49  52  50  00014 P.AAD:  .ASCII   \PRIMARY\<0>                                   ;
                                           010E0007  0001C P.AAC:  .LONG    17694727                                       ;
                                          00000000'  00020         .ADDRESS P.AAD                                          ;
                     00  54  4C  55  41  46  45  44  00024 P.AAF:  .ASCII   \DEFAULT\<0>                                   ;
                                           010E0007  0002C P.AAE:  .LONG    17694727                                       ;
                                          00000000'  00030         .ADDRESS P.AAF                                          ;

                                                                   .EXTRN   LIB$CVT_DTB, CLI$GET_VALUE
                                                                   .EXTRN   CLI$PRESENT, EXE$GL_FLAGS
                                                                   .EXTRN   CTL$GL_PCB, CTL$GL_PHD
                                                                   .EXTRN   CTL$GQ_PROCPRIV
                                                                   .EXTRN   SYS$GL_JOBCTLMB
                                                                   .EXTRN   SYS$GW_IJOBCNT, SYS$GW_IJOBLIM
                                                                   .EXTRN   SYS$GB_DFMBC, PIO$GB_DFMBC
                                                                   .EXTRN   SYS$GB_DFNBC, PIO$GB_DFNBC
                                                                   .EXTRN   SYS$GB_RMSPROLOG
                                                                   .EXTRN   PIO$GB_RMSPROLOG
                                                                   .EXTRN   SYS$GW_RMSEXTEND
                                                                   .EXTRN   PIO$GW_RMSEXTEND
                                                                   .EXTRN   SYS$GB_DFMBFSDK
                                                                   .EXTRN   SYS$GB_DFMBFSMT
                                                                   .EXTRN   SYS$GB_DFMBFSUR
                                                                   .EXTRN   SYS$GB_DFMBFIDX
                                                                   .EXTRN   SYS$GB_DFMBFHSH
                                                                   .EXTRN   SYS$GB_DFMBFREL
                                                                   .EXTRN   PIO$GB_DFMBFSDK
                                                                   .EXTRN   PIO$GB_DFMBFSMT
                                                                   .EXTRN   PIO$GB_DFMBFSUR
                                                                   .EXTRN   PIO$GB_DFMBFIDX
                                                                   .EXTRN   PIO$GB_DFMBFHSH
                                                                   .EXTRN   PIO$GB_DFMBFREL
                                                                   .EXTRN   EXE$V_EXPLICITP
                                                                   .EXTRN   EXE$V_EXPLICITS
                                                                   .EXTRN   CLI$_ABSENT, SET$_NEWLIMS
                                                                   .EXTRN   SET$_INTSET, SYS$CMKRNL

                                                                   .PSECT   $CODE$,NOWRT,2

                                                001C 00000         .ENTRY   SET$DAY, Save R2,R3,R4                         ; 0211
                          54 00000000G   00   9E 00002         MOVAB    LIB$STOP, R4                                   ;
                          53 00000000'   EF   9E 00009         MOVAB    P.AAA, R3                                      ;
                          52 00000000G   00   9E 00010         MOVAB    CLI$PRESENT, R2                                ;
                          5E             0C   C2 00017         SUBL2    #12, SP                                        ;
                                         6E   94 0001A         CLRB     FLAGS                                          ; 0212
                                         53   DD 0001C         PUSHL    R3                                             ; 0237
                          62             01   FB 0001E         CALLS    #1, CLI$PRESENT                                ;
   6E           01        03             50   F0 00021         INSV     R0, #3, #1, FLAGS                              ;
                                    10   A3   9F 00026         PUSHAB   P.AAC                                          ; 0238
                          62             01   FB 00029         CALLS    #1, CLI$PRESENT                                ;
   6E           01        02             50   F0 0002C         INSV     R0, #2, #1, FLAGS                              ;
                                    20   A3   9F 00031         PUSHAB   P.AAE                                          ; 0239
                          62             01   FB 00034         CALLS    #1, CLI$PRESENT                                ;
   6E           01        04             50   F0 00037         INSV     R0, #4, #1, FLAGS                              ;
                08 00000000G  00             02   E0 0003C         BBS      #2, CTL$GQ_PROCPRIV+2, 1$                      ; 0244
                          7E           2894   8F   3C 00044         MOVZWL   #10388, -(SP)                                  ; 0245
                          64             01   FB 00049         CALLS    #1, LIB$STOP                                   ;
                     04   AE             01   D0 0004C 1$:     MOVL     #1, ARGLST                                     ; 0250
```

```
                 08    AE              6E  9E 00050         MOVAB    FLAGS, ARGLST+4                          ; 0251
                                 04    AE  9F 00054         PUSHAB   ARGLST                                   ; 0253
                          00000000V    EF  9F 00057         PUSHAB   SETDAYKNL                                ;
         00000000G  00                 02  FB 0005D         CALLS    #2, SYS$CMKRNL                           ;
                    05                 50  E8 00064         BLBS     STATUS, 2$                               ;
                                       50  DD 00067         PUSHL    STATUS                                   ; 0254
                    64                 01  FB 00069         CALLS    #1, LIB$STOP                             ;
                                           04 0006C 2$:     RET                                               ; 0257

; Routine Size:  109 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 ROUTINE setdayknl (flags) =
2 BEGIN
2 !++
2 !
2 ! This routine executes in kernel mode, setting the longword
2 ! EXE$GL_FLAGS to signify what kind of day it is.
2 !
2 ! Inputs:
2 !       FLAGS - address of the flags byte.
2 !
2 ! Outputs:
2 !       None.
2 !
2 !--
2
2 MAP flags : REF $BBLOCK;
2
2 !
2 ! If the day is to be set primary, then turn off the EXPLICITP bit and
2 ! turn on the EXPLICITS bit.
2 !
2 IF .flags[set$v_primary]
2 THEN
3     BEGIN
3     exe$gl_flags[0, exe$v_explicitp, 1, 0] = 0;
3     exe$gl_flags[0, exe$v_explicits, 1, 0] = 1;
3     END
3
3 !
3 ! If not primary, check to see if the day should be set secondary.
3 !
2 ELSE
3     BEGIN
3     IF .flags[set$v_secondary]
3     THEN
4         BEGIN
4         exe$gl_flags[0, exe$v_explicitp, 1, 0] = 1;
4         exe$gl_flags[0, exe$v_explicits, 1, 0] = 1;
4         END
4
4 !
4 ! If set to be /DEFAULT, then do it.
4 !
3     ELSE
4         BEGIN
4         IF .flags[set$v_default]
4         THEN exe$gl_flags[0, exe$v_explicitp, 1, 0] = 0;
3         END;
2     END;
2
2 RETURN 1;
1 END;
```

```
                               000C 00000 SETDAYKNL:
                                                   .WORD    Save R2,R3                                  ; 0258
             53 00000000G   8F   D0 00002          MOVL     #EXE$V_EXPLICITP, R3                        ;
             52 00000000G   00   9E 00009          MOVAB    EXE$GL_FLAGS, R2                            ;
06      04   BC             02   E1 00010          BBC      #2, @FLAGS, 1$                              ; 0279
0B           62             53   E5 00015          BBCC     R3, EXE$GL_FLAGS, 2$                        ; 0282
                            09   11 00019          BRB      2$                                          ; 0283
0E      04   BC             03   E1 0001B 1$:      BBC      #3, @FLAGS, 3$                              ; 0291
00           62             53   E2 00020          BBSS     R3, EXE$GL_FLAGS, 2$                        ; 0294
0B           62 00000000G   8F   E2 00024 2$:      BBSS     #EXE$V_EXPLICITS, EXE$GL_FLAGS, 4$          ; 0295
                            09   11 0002C          BRB      4$                                          ; 0291
04      04   BC             04   E1 0002E 3$:      BBC      #4, @FLAGS, 4$                              ; 0303
00           62             53   E5 00033          BBCC     R3, EXE$GL_FLAGS, 4$                        ; 0304
             50             01   D0 00037 4$:      MOVL     #1, R0                                      ; 0308
                                 04 0003A          RET                                                  ; 0309

; Routine Size:  59 bytes,    Routine Base:  $CODE$ + 006D
```

```
   1 GLOBAL ROUTINE set$login : NOVALUE =
   2 BEGIN
   2 !++
   2 !
   2 ! This routine sets the number of interactive logins permitted.
   2 !
   2 ! Inputs:
   2 !       None.  The CLI is interrogated for the number.
   2 !
   2 ! Outputs:
   2 !       None.
   2 !
   2 !
   2 !--
   2
   2 LOCAL
   2     status,                              ! General status return
   2     number,                              ! Number of users
   2     arglst : VECTOR[2],                  ! Argument list for $CMKRNL call
   2     desc : $BBLOCK[dsc$c_s_bln];         ! Descriptor to get number
   2
   2 !
   2 ! If the user doesn't have OPER, don't allow the operation.
   2 !
   2 IF NOT .ctl$gq_procpriv[prv$v_oper]
   2 THEN SIGNAL_STOP(ss$_nooper);
   2
   2 !
   2 ! Get the number of users.
   2 !
   2 $init_dyndesc(desc);                     ! Make the descriptor dynamic
   2 IF cli$present(%ASCID 'INTERACTIVE')
   2 THEN
   2     cli$get_value(%ASCID 'INTERACTIVE', ! Get the number
   2                   desc);
   2
   2
   2 !
   2 ! If the number is non-zero, go set it.
   2 !
   2 IF .desc[dsc$w_length] NEQ 0
   2 THEN
   3     BEGIN
   4     IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
   4                                  .desc[dsc$a_pointer],
   4                                 number))
   3     THEN
   4         BEGIN
   4         SIGNAL(set$_valerr);
   4         RETURN;
   3         END;
   3     arglst[0] = 1;
   3     arglst[1] = .number;
P  4     IF NOT (status = $CMKRNL(ROUTIN = setlogknl,
   4                              ARGLST = arglst))
   3     THEN
   4         BEGIN
```

```
;  372      0367 4          SIGNAL(.status);
;  373      0368 4          RETURN;
;  374      0369 3          END;
;  375      0370 2      END;
;  376      0371 2
;  377      0372 2  !
;  378      0373 2  ! If we get this far, then use SIGNAL to output the current interactive
;  379      0374 2  ! limit.
;  380      0375 2  !
;  381      0376 2  SIGNAL(set$_intset, 2, .sys$gw_ijoblim, .sys$gw_ijobcnt);
;  382      0377 2  RETURN 1;
;  383      0378 1  END;


                                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

       00  45  56  49  54  43  41  52  45  54  4E  49  00034 P.AAH:   .ASCII  \INTERACTIVE\<0>              ;
                                         010E000B  00040 P.AAG:   .LONG   17694731                      :
                                        00000000'  00044          .ADDRESS P.AAH                        :
       00  45  56  49  54  43  41  52  45  54  4E  49  00048 P.AAJ:   .ASCII  \INTERACTIVE\<0>              :
                                         010E000B  00054 P.AAI:   .LONG   17694731                      :
                                        00000000'  00058          .ADDRESS P.AAJ                        :


                                                                              .PSECT  $CODE$,NOWRT,2

                                           000C 00000          .ENTRY  SET$LOGIN, Save R2,R3           ; 0310
                      53 00000000G    00     9E 00002          MOVAB   LIB$SIGNAL, R3                  :
                      5E              14     C2 00009          SUBL2   #20, SP                         :
           0C 00000000G 00            02     E0 0000C          BBS     #2, CTL$GQ_PROCPRIV+2, 1$       ; 0334
                      7E     2894     8F     3C 00014          MOVZWL  #10388, -(SP)                   ; 0335
              00000000G 00            01     FB 00019          CALLS   #1, LIB$STOP                    ; 0335
                 04   AE 020E0000     8F     D0 00020 1$:      MOVL    #34471936, DESC                 ; 0340
                             08       AE     D4 00028          CLRL    DESC+4                          :
                        00000000'     EF     9F 0002B          PUSHAB  P.AAG                           ; 0341
              00000000G 00            01     FB 00031          CALLS   #1, CLI$PRESENT                 :
                      10              50     E9 00038          BLBC    R0, 2$                          :
                             04       AE     9F 0003B          PUSHAB  DESC                            ; 0343
                        00000000'     EF     9F 0003E          PUSHAB  P.AAI                           :
              00000000G 00            02     FB 00044          CALLS   #2, CLI$GET_VALUE               :
                             04       AE     B5 0004B 2$:      TSTW    DESC                            ; 0350
                                      42     13 0004E          BEQL    5$                              :
                                      5E     DD 00050          PUSHL   SP                              ; 0353
                             0C       AE     DD 00052          PUSHL   DESC+4                          ; 0354
                      7E     0C       AE     3C 00055          MOVZWL  DESC, -(SP)                     ; 0353
              00000000G 00            03     FB 00059          CALLS   #3, LIB$CVT_DTB                 :
                      52              50     D0 00060          MOVL    R0, STATUS                      :
                      08              52     E8 00063          BLBS    STATUS, 3$                      :
                        007711EA      8F     DD 00066          PUSHL   #7803370                        ; 0358
                                      20     11 0006C          BRB     4$                              :
                 0C   AE              01     D0 0006E 3$:      MOVL    #1, ARGLST                      ; 0361
                 10   AE              6E     D0 00072          MOVL    NUMBER, ARGLST+4                ; 0362
                             0C       AE     9F 00076          PUSHAB  ARGLST                          ; 0364
                        00000000V     EF     9F 00079          PUSHAB  SETLOGKNL                       :
              00000000G 00            02     FB 0007F          CALLS   #2, SYS$CMKRNL                  :
                52                 50  D0 00086      MOVL    R0, STATUS                          ;
                06                 52  E8 00089      BLBS    STATUS, 5$                          ;
                                   52  DD 0008C      PUSHL   STATUS                              ; 0367
                63                 01  FB 0008E 4$:  CALLS   #1, LIB$SIGNAL                      ;
                                   04  00091         RET                                         ; 0366
                7E 00000000G       00  3C 00092 5$:  MOVZWL  SYS$GW_IJOBCNT, -(SP)               ; 0376
                7E 00000000G       00  3C 00099      MOVZWL  SYS$GW_IJOBLIM, -(SP)               ;
                                   02  DD 000A0      PUSHL   #2                                  ;
                   00000000G       8F  DD 000A2      PUSHL   #SET$_INTSET                        ;
                63                 04  FB 000A8      CALLS   #4, LIB$SIGNAL                      ;
                                   04  000AB         RET                                         ; 0378

; Routine Size:  172 bytes,    Routine Base:  $CODE$ + 00A8
;  385          0379  1 ROUTINE setlogknl (number) =
;  386          0380  2 BEGIN
;  387          0381  2 !++
;  388          0382  2 !
;  389          0383  2 ! This routine is called in kernel mode to set the number of interactive
;  390          0384  2 ! processes.
;  391          0385  2 !
;  392          0386  2 ! Inputs:
;  393          0387  2 !       NUMBER - address of the limit to set.
;  394          0388  2 !
;  395          0389  2 ! Outputs:
;  396          0390  2 !       None.  The interactive job count limit is set.
;  397          0391  2 !
;  398          0392  2 !--
;  399          0393  2
;  400          0394  2 !
;  401          0395  2 ! Set the job limit.
;  402          0396  2 !
;  403          0397  2 sys$gw_ijoblim = .number;
;  404          0398  2
;  405          0399  2 !
;  406          0400  2 ! If the limit is non-zero, turn on interactive jobs.  This is done by
;  407          0401  2 ! clearing the high bit of the job controller mailbox status word.
;  408          0402  2 !
;  409          0403  2 IF .number NEQ 0                 ! If at least one allowed to login,
;  410          0404  2 THEN sys$gl_jobctlmb[ucb$v_tt_nologins] = 0;    ! enable interactive prompts.
;  411          0405  2
;  412          0406  2 RETURN 1;
;  413          0407  1 END;



                                          0000 00000 SETLOGKNL:
                                                          .WORD    Save nothing                        ; 0379
                 00000000G  00        04    AC   B0 00002          MOVW     NUMBER, SYS$GW_IJOBLIM              ; 0397
                                      04    AC   D5 0000A          TSTL     NUMBER                              ; 0403
                                            08   13 0000D          BEQL     1$
                 00000000G  00        80    8F   8A 0000F          BICB2    #128, SYS$GL_JOBCTLMB+105           ; 0404
                            50              01   D0 00017 1$:      MOVL     #1, R0                              ; 0406
                                                 04 0001A          RET                                          ; 0407

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 0154
```

```
1 GLOBAL ROUTINE set$rms_default : NOVALUE =
2 BEGIN
2 !++
2 !
2 ! This routine implements the SET RMS_DEFAULT command.  The values and
2 ! qualifiers are collected and checked, then a kernel call is made to
2 ! actually set the parameters.  In order to change RMS defaults for the
2 ! system, the process must have CMKRNL privilege.
2 !
2 ! Inputs:
2 !       None.  The CLI is interrogated.
2 !
2 ! Outputs:
2 !       None.  The RMS defaults are changed.
2 !
2 !--
2
2 LOCAL
2     status,                                       ! General status return
2     block_count,                                  ! Block count
2     buffer_count,                                 ! Buffer count
2     net_block_count,                              ! Network Block count
2     prolog,                                       ! Prolog level
2     extend,                                       ! Extend quantity
2     desc : $BBLOCK[dsc$c_s_bln],                  ! General descriptor
2     arglst : VECTOR[6],                           ! Argument list for CMKRNL call
2     flags : $BBLOCK[4] INITIAL(0);                ! Flags longword
2
2 !
2 ! First, get the qualifiers and quantities.
2 !
2 $init_dyndesc(desc);                              ! Make the descriptor dynamic
2
2 !
2 ! Get the block count.  If there, convert it to a number.
2 !
3 IF (flags[set$v_block] = cli$present(%ASCID 'BLOCK_COUNT'))
2 THEN
2     IF cli$get_value(%ASCID 'BLOCK_COUNT', desc)
2     THEN
3         BEGIN
4         IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
4                                      .desc[dsc$a_pointer],
4                                      block_count))
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
3         IF .block_count GTR 127                   ! Check for in range
3         OR .block_count LSS 0
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
2         END;
```

```
2 !
2 ! Get the network block count.  If there, convert it to a number.
2 !
3 IF (flags[set$v_netblk] = cli$present(%ASCID 'NETWORK_BLOCK_COUNT'))
2 THEN
2     IF cli$get_value(%ASCID 'NETWORK_BLOCK_COUNT', desc)
2     THEN
3         BEGIN
4         IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
4                                      .desc[dsc$a_pointer],
4                                      net_block_count))
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
3         IF .net_block_count GTR 127                        ! Check for in range
3         OR .net_block_count LSS 0
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
2         END;
2
2 !
2 ! Get the buffer count.  If there, convert to a number.
2 !
3 IF (flags[set$v_buffer] = cli$present(%ASCID 'BUFFER_COUNT'))
2 THEN
2     IF cli$get_value(%ASCID 'BUFFER_COUNT', desc)
2     THEN
3         BEGIN
4         IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
4                                      .desc[dsc$a_pointer],
4                                      buffer_count))
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
3         IF .buffer_count GTR 127                           ! Check for in range
3         OR .buffer_count LSS -127
3         THEN
4             BEGIN
4             SIGNAL(set$_valerr);
4             RETURN;
3             END;
2         END;
2
2 !
2 ! Get the prologue level.  If there, convert to a number.
2 !
3 IF (flags[set$v_prolog] = cli$present(%ASCID 'PROLOGUE'))
2 THEN
2     IF cli$get_value(%ASCID 'PROLOGUE', desc)
2     THEN
```

```
3           BEGIN
4           IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
4                                         .desc[dsc$a_pointer],
4                                         prolog))
3           THEN
4               BEGIN
4               SIGNAL(set$_valerr);
4               RETURN;
3               END;
4           IF NOT (.prolog EQL 0 OR                            ! Check for valid prolog level
4                   .prolog EQL 2 OR
4                   .prolog EQL 3)
3           THEN
4               BEGIN
4               SIGNAL(set$_valerr);
4               RETURN;
3               END;
2           END;
2
2   !
2   ! Get the extend quantity.  If there, convert it to a number.
2   !
3   IF (flags[set$v_extend] = cli$present(%ASCID 'EXTEND_QUANTITY'))
2   THEN
2       IF cli$get_value(%ASCID 'EXTEND_QUANTITY', desc)
2       THEN
3           BEGIN
4           IF NOT (status = lib$cvt_dtb(.desc[dsc$w_length],
4                                         .desc[dsc$a_pointer],
4                                         extend))
3           THEN
4               BEGIN
4               SIGNAL(set$_valerr);
4               RETURN;
3               END;
3           IF .extend GTR 65535                        ! Check for in range
3           OR .extend LSS 0
3           THEN
4               BEGIN
4               SIGNAL(set$_valerr);
4               RETURN;
3               END;
2           END;
2
2   !
2   ! Now to collect all the qualifiers
2   !
2   flags[set$v_hash] = cli$present(%ASCID 'HASH');
2   flags[set$v_index] = cli$present(%ASCID 'INDEXED');
2   flags[set$v_rel] = cli$present(%ASCID 'RELATIVE');
2   flags[set$v_disk] = cli$present(%ASCID 'DISK');
2   flags[set$v_tape] = cli$present(%ASCID 'MAGTAPE');
2   flags[set$v_unit] = cli$present(%ASCID 'UNIT_RECORD');
2   flags[set$v_system] = cli$present(%ASCID 'SYSTEM');
2
2   !
2   ! If /SEQUENTIAL was specified, then turn it on for all sequential
```

```
;  586        0579  2 ! devices, ie. disk, magtape, and unit_record.
;  587        0580  2 !
;  588        0581  2 IF cli$present(%ASCID 'SEQUENTIAL')               ! If /SEQUENTIAL,
;  589        0582  2 THEN flags[set$v_seq] = flags[set$v_disk]         ! turn them all on
;  590        0583  2                       = flags[set$v_tape]
;  591        0584  2                       = flags[set$v_unit]
;  592        0585  2                       = 1;
;  593        0586  2
;  594        0587  2 !
;  595        0588  2 ! The SET RMS command defaults to /MAGTAPE/DISK/UNIT if no qualifiers are
;  596        0589  2 ! specified.  Do that manually.
;  597        0590  2 !
;  598        0591  3 IF NOT (.flags[set$v_tape] OR                      ! If nothing turned on,
;  599        0592  3         .flags[set$v_disk] OR
;  600        0593  3         .flags[set$v_unit] OR
;  601        0594  3         .flags[set$v_index] OR
;  602        0595  3         .flags[set$v_rel])
;  603        0596  2 THEN flags[set$v_disk] = flags[set$v_tape]        ! turn on disk, tape, and
;  604        0597  2                        = flags[set$v_unit]        ! unit record
;  605        0598  2                        = 1;
;  606        0599  2
;  607        0600  2 !
;  608        0601  2 ! If /SYSTEM was specified, check that the user has CMKRNL privilege.
;  609        0602  2 ! Otherwise, reject the request.
;  610        0603  2 !
;  611        0604  2 IF .flags[set$v_system]
;  612        0605  2 THEN
;  613        0606  3     BEGIN
;  614        0607  3     IF NOT .ctl$gq_procpriv[prv$v_cmkrnl]
;  615        0608  3     THEN
;  616        0609  4         BEGIN
;  617        0610  4         SIGNAL(ss$_nocmkrnl);
;  618        0611  4         RETURN;
;  619        0612  3         END;
;  620        0613  2     END;
;  621        0614  2
;  622        0615  2 !
;  623        0616  2 ! Build the argument list and call the kernel mode routine that will actually
;  624        0617  2 ! do what is requested.
;  625        0618  2 !
;  626        0619  2 arglst[0] = 6;
;  627        0620  2 arglst[1] = flags;
;  628        0621  2 arglst[2] = .block_count;
;  629        0622  2 arglst[3] = .buffer_count;
;  630        0623  2 arglst[4] = .prolog;
;  631        0624  2 arglst[5] = .extend;
;  632        0625  2 arglst[6] = .net_block_count;
;  633      P 0626  3 IF NOT (status = $CMKRNL(ROUTIN = setrmsknl,
;  634        0627  3                          ARGLST = arglst))
;  635        0628  2 THEN SIGNAL(.status);
;  636        0629  2
;  637        0630  2 RETURN;
;  638        0631  1 END;


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

```
               00 54 4E 55 4F 43 5F 4B 43 4F 4C 42  0005C P.AAL:  .ASCII   \BLOCK_COUNT\<0>
                                        010E000B  00068 P.AAK:  .LONG    17694731
                                        00000000' 0006C         .ADDRESS P.AAL
               00 54 4E 55 4F 43 5F 4B 43 4F 4C 42  00070 P.AAN:  .ASCII   \BLOCK_COUNT\<0>
                                        010E000B  0007C P.AAM:  .LONG    17694731
                                        00000000' 00080         .ADDRESS P.AAN
43 5F 4B 43 4F 4C 42 5F 4B 52 4F 57 54 45 4E  00084 P.AAP:  .ASCII   \NETWORK_BLOCK_COUNT\<0>
                                00 54 4E 55 4F  00093
                                        010E0013  00098 P.AAO:  .LONG    17694739
                                        00000000' 0009C         .ADDRESS P.AAP
43 5F 4B 43 4F 4C 42 5F 4B 52 4F 57 54 45 4E  000A0 P.AAR:  .ASCII   \NETWORK_BLOCK_COUNT\<0>
                                00 54 4E 55 4F  000AF
                                        010E0013  000B4 P.AAQ:  .LONG    17694739
                                        00000000' 000B8         .ADDRESS P.AAR
               54 4E 55 4F 43 5F 52 45 46 46 55 42  000BC P.AAT:  .ASCII   \BUFFER_COUNT\
                                        010E000C  000C8 P.AAS:  .LONG    17694732
                                        00000000' 000CC         .ADDRESS P.AAT
               54 4E 55 4F 43 5F 52 45 46 46 55 42  000D0 P.AAV:  .ASCII   \BUFFER_COUNT\
                                        010E000C  000DC P.AAU:  .LONG    17694732
                                        00000000' 000E0         .ADDRESS P.AAV
                           45 55 47 4F 4C 4F 52 50  000E4 P.AAX:  .ASCII   \PROLOGUE\
                                        010E0008  000EC P.AAW:  .LONG    17694728
                                        00000000' 000F0         .ADDRESS P.AAX
                           45 55 47 4F 4C 4F 52 50  000F4 P.AAZ:  .ASCII   \PROLOGUE\
                                        010E0008  000FC P.AAY:  .LONG    17694728
                                        00000000' 00100         .ADDRESS P.AAZ
59 54 49 54 4E 41 55 51 5F 44 4E 45 54 58 45  00104 P.ABB:  .ASCII   \EXTEND_QUANTITY\<0>
                                              00  00113
                                        010E000F  00114 P.ABA:  .LONG    17694735
                                        00000000' 00118         .ADDRESS P.ABB
59 54 49 54 4E 41 55 51 5F 44 4E 45 54 58 45  0011C P.ABD:  .ASCII   \EXTEND_QUANTITY\<0>
                                              00  0012B
                                        010E000F  0012C P.ABC:  .LONG    17694735
                                        00000000' 00130         .ADDRESS P.ABD
                                       48 53 41 48  00134 P.ABF:  .ASCII   \HASH\
                                        010E0004  00138 P.ABE:  .LONG    17694724
                                        00000000' 0013C         .ADDRESS P.ABF
                           00 44 45 58 45 44 4E 49  00140 P.ABH:  .ASCII   \INDEXED\<0>
                                        010E0007  00148 P.ABG:  .LONG    17694727
                                        00000000' 0014C         .ADDRESS P.ABH
                           45 56 49 54 41 4C 45 52  00150 P.ABJ:  .ASCII   \RELATIVE\
                                        010E0008  00158 P.ABI:  .LONG    17694728
                                        00000000' 0015C         .ADDRESS P.ABJ
                                       4B 53 49 44  00160 P.ABL:  .ASCII   \DISK\
                                        010E0004  00164 P.ABK:  .LONG    17694724
                                        00000000' 00168         .ADDRESS P.ABL
                           00 45 50 41 54 47 41 4D  0016C P.ABN:  .ASCII   \MAGTAPE\<0>
                                        010E0007  00174 P.ABM:  .LONG    17694727
                                        00000000' 00178         .ADDRESS P.ABN
               00 44 52 4F 43 45 52 5F 54 49 4E 55  0017C P.ABP:  .ASCII   \UNIT_RECORD\<0>
                                        010E000B  00188 P.ABO:  .LONG    17694731
                                        00000000' 0018C         .ADDRESS P.ABP
                           00 00 4D 45 54 53 59 53  00190 P.ABR:  .ASCII   \SYSTEM\<0><0>
                                        010E0006  00198 P.ABQ:  .LONG    17694726
                                        00000000' 0019C         .ADDRESS P.ABR
               00 00 4C 41 49 54 4E 45 55 51 45 53  001A0 P.ABT:  .ASCII   \SEQUENTIAL\<0><0>
```

```
                                010E000A  001AC P.ABS:  .LONG    17694730                            ;
                                00000000' 001B0         .ADDRESS P.ABT                               ;

                                                        .PSECT   $CODE$,NOWRT,2

                                     007C 00000         .ENTRY   SET$RMS_DEFAULT, Save R2,R3,R4,R5,R6  ; 0408
                    56 00000000G   00   9E 00002         MOVAB    LIB$CVT_DTB, R6                      ;
                    55 00000000G   00   9E 00009         MOVAB    CLI$GET_VALUE, R5                    ;
                    54 00000000G   00   9E 00010         MOVAB    CLI$PRESENT, R4                      ;
                    53 00000000'   EF   9E 00017         MOVAB    P.AAK, R3                            ;
                    5E             38   C2 0001E         SUBL2    #56, SP                              ;
                              14   AE   D4 00021         CLRL     FLAGS                                ; 0409
            30   AE 020E0000       8F   D0 00024         MOVL     #34471936, DESC                      ; 0439
                              34   AE   D4 0002C         CLRL     DESC+4                               ;
                                   53   DD 0002F         PUSHL    R3                                   ; 0444
                    64             01   FB 00031         CALLS    #1, CLI$PRESENT                      ;
 14   AE         01 03             50   F0 00034         INSV     R0, #3, #1, FLAGS                    ;
                    2B             50   E9 0003A         BLBC     R0, 1$                               ;
                              30   AE   9F 0003D         PUSHAB   DESC                                 ; 0446
                              14   A3   9F 00040         PUSHAB   P.AAM                                ;
                    65             02   FB 00043         CALLS    #2, CLI$GET_VALUE                    ;
                    1F             50   E9 00046         BLBC     R0, 1$                               ;
                                   5E   DD 00049         PUSHL    SP                                   ; 0449
                              38   AE   DD 0004B         PUSHL    DESC+4                               ; 0450
                    7E        38   AE   3C 0004E         MOVZWL   DESC, -(SP)                          ; 0449
                    66             03   FB 00052         CALLS    #3, LIB$CVT_DTB                      ;
                    52             50   D0 00055         MOVL     R0, STATUS                           ;
                    75             52   E9 00058         BLBC     STATUS, 3$                           ;
          0000007F  8F             6E   D1 0005B         CMPL     BLOCK_COUNT, #127                    ; 0457
                                   77   14 00062         BGTR     4$                                   ;
                                   6E   D5 00064         TSTL     BLOCK_COUNT                          ; 0458
                                   7D   19 00066         BLSS     5$                                   ;
                              30   A3   9F 00068 1$:     PUSHAB   P.AAO                                ; 0468
                    64             01   FB 0006B         CALLS    #1, CLI$PRESENT                      ;
 15   AE         01 06             50   F0 0006E         INSV     R0, #6, #1, FLAGS+1                  ;
                    2E             50   E9 00074         BLBC     R0, 2$                               ;
                              30   AE   9F 00077         PUSHAB   DESC                                 ; 0470
                              4C   A3   9F 0007A         PUSHAB   P.AAQ                                ;
                    65             02   FB 0007D         CALLS    #2, CLI$GET_VALUE                    ;
                    22             50   E9 00080         BLBC     R0, 2$                               ;
                              04   AE   9F 00083         PUSHAB   NET_BLOCK_COUNT                      ; 0473
                              38   AE   DD 00086         PUSHL    DESC+4                               ; 0474
                    7E        38   AE   3C 00089         MOVZWL   DESC, -(SP)                          ; 0473
                    66             03   FB 0008D         CALLS    #3, LIB$CVT_DTB                      ;
                    52             50   D0 00090         MOVL     R0, STATUS                           ;
                    7E             52   E9 00093         BLBC     STATUS, 7$                           ;
          0000007F  8F        04   AE   D1 00096         CMPL     NET_BLOCK_COUNT, #127                ; 0481
                                   3B   14 0009E         BGTR     4$                                   ;
                              04   AE   D5 000A0         TSTL     NET_BLOCK_COUNT                      ; 0482
                                   40   19 000A3         BLSS     5$                                   ;
                              60   A3   9F 000A5 2$:     PUSHAB   P.AAS                                ; 0493
                    64             01   FB 000A8         CALLS    #1, CLI$PRESENT                      ;
 14   AE         01 04             50   F0 000AB         INSV     R0, #4, #1, FLAGS                    ;
                    33             50   E9 000B1         BLBC     R0, 6$                               ;
                              30   AE   9F 000B4         PUSHAB   DESC                                 ; 0495
```

```
                                    74  A3  9F 000B7          PUSHAB  P.AAU                          ;
                          65        02  FB 000BA          CALLS   #2, CLI$GET_VALUE              ;
                          27        50  E9 000BD          BLBC    R0, 6$                         ;
                                08  AE  9F 000C0          PUSHAB  BUFFER_COUNT                   ; 0498
                                38  AE  DD 000C3          PUSHL   DESC+4                         ; 0499
                          7E    38  AE  3C 000C6          MOVZWL  DESC, -(SP)                    ; 0498
                          66        03  FB 000CA          CALLS   #3, LIB$CVT_DTB                ;
                          52        50  D0 000CD          MOVL    R0, STATUS                     ;
                          41        52  E9 000D0  3$:     BLBC    STATUS, 7$                     ;
              0000007F    8F    08  AE  D1 000D3          CMPL    BUFFER_COUNT, #127             ; 0506
                                    48  14 000DB  4$:     BGTR    8$                             ;
              FFFFFF81    8F    08  AE  D1 000DD          CMPL    BUFFER_COUNT, #-127            ; 0507
                                    7F  19 000E5  5$:     BLSS    10$                            ;
                              0084  C3  9F 000E7  6$:     PUSHAB  P.AAW                          ; 0518
                          64        01  FB 000EB          CALLS   #1, CLI$PRESENT                ;
 14  AE             01    05        50  F0 000EE          INSV    R0, #5, #1, FLAGS              ;
                          30        50  E9 000F4          BLBC    R0, 9$                         ;
                                30  AE  9F 000F7          PUSHAB  DESC                           ; 0520
                              0094  C3  9F 000FA          PUSHAB  P.AAY                          ;
                          65        02  FB 000FE          CALLS   #2, CLI$GET_VALUE              ;
                          23        50  E9 00101          BLBC    R0, 9$                         ;
                                0C  AE  9F 00104          PUSHAB  PROLOG                         ; 0523
                                38  AE  DD 00107          PUSHL   DESC+4                         ; 0524
                          7E    38  AE  3C 0010A          MOVZWL  DESC, -(SP)                    ; 0523
                          66        03  FB 0010E          CALLS   #3, LIB$CVT_DTB                ;
                          52        50  D0 00111          MOVL    R0, STATUS                     ;
                          4F        52  E9 00114  7$:     BLBC    STATUS, 10$                    ;
                          50    0C  AE  D0 00117          MOVL    PROLOG, R0                     ; 0531
                                    0A  13 0011B          BEQL    9$                             ;
                                02  50  D1 0011D          CMPL    R0, #2                         ; 0532
                                    05  13 00120          BEQL    9$                             ;
                                03  50  D1 00122          CMPL    R0, #3                         ; 0533
                                    3F  12 00125  8$:     BNEQ    10$                            ;
                              00AC  C3  9F 00127  9$:     PUSHAB  P.ABA                          ; 0544
                          64        01  FB 0012B          CALLS   #1, CLI$PRESENT                ;
 15  AE             01    05        50  F0 0012E          INSV    R0, #5, #1, FLAGS+1            ;
                          38        50  E9 00134          BLBC    R0, 11$                        ;
                                30  AE  9F 00137          PUSHAB  DESC                           ; 0546
                              00C4  C3  9F 0013A          PUSHAB  P.ABC                          ;
                          65        02  FB 0013E          CALLS   #2, CLI$GET_VALUE              ;
                          2B        50  E9 00141          BLBC    R0, 11$                        ;
                                10  AE  9F 00144          PUSHAB  EXTEND                         ; 0549
                                38  AE  DD 00147          PUSHL   DESC+4                         ; 0550
                          7E    38  AE  3C 0014A          MOVZWL  DESC, -(SP)                    ; 0549
                          66        03  FB 0014E          CALLS   #3, LIB$CVT_DTB                ;
                          52        50  D0 00151          MOVL    R0, STATUS                     ;
                          0F        52  E9 00154          BLBC    STATUS, 10$                    ;
              0000FFFF    8F    10  AE  D1 00157          CMPL    EXTEND, #65535                 ; 0557
                                    05  14 0015F          BGTR    10$                            ;
                                10  AE  D5 00161          TSTL    EXTEND                         ; 0558
                                    09  18 00164          BGEQ    11$                            ;
                          007711EA  8F  DD 00166  10$:    PUSHL   #7803370                       ; 0561
                                  00D0  31 0016C          BRW     15$                            ;
                              00D0  C3  9F 0016F  11$:    PUSHAB  P.ABE                          ; 0569
                          64        01  FB 00173          CALLS   #1, CLI$PRESENT                ;
 15  AE             01    04        50  F0 00176          INSV    R0, #4, #1, FLAGS+1            ;
                              00E0  C3  9F 0017C          PUSHAB  P.ABG                          ; 0570
```

```
SETMISC                                                  G 12
V04-000                                                  16-Sep-1984 00:43:54    VAX-11 Bliss-32 V4.0-742         Page  22
                                                         14-Sep-1984 12:09:11    [CLIUTL.SRC]SETMISC.B32;1              (8)

                            64             01 FB 00180       CALLS   #1, CLI$PRESENT                ;
     15 AE        01        03             50 F0 00183       INSV    R0, #3, #1, FLAGS+1            ;
                                      00F0 C3 9F 00189       PUSHAB  P.ABI                          ; 0571
                            64             01 FB 0018D       CALLS   #1, CLI$PRESENT                ;
     15 AE        01        02             50 F0 00190       INSV    R0, #2, #1, FLAGS+1            ;
                                      00FC C3 9F 00196       PUSHAB  P.ABK                          ; 0572
                            64             01 FB 0019A       CALLS   #1, CLI$PRESENT                ;
     14 AE        01        06             50 F0 0019D       INSV    R0, #6, #1, FLAGS              ;
                                      010C C3 9F 001A3       PUSHAB  P.ABM                          ; 0573
                            64             01 FB 001A7       CALLS   #1, CLI$PRESENT                ;
     14 AE        01        07             50 F0 001AA       INSV    R0, #7, #1, FLAGS              ;
                                      0120 C3 9F 001B0       PUSHAB  P.ABO                          ; 0574
                            64             01 FB 001B4       CALLS   #1, CLI$PRESENT                ;
     15 AE        01        00             50 F0 001B7       INSV    R0, #0, #1, FLAGS+1            ;
                                      0130 C3 9F 001BD       PUSHAB  P.ABQ                          ; 0575
                            64             01 FB 001C1       CALLS   #1, CLI$PRESENT                ;
     14 AE        01        02             50 F0 001C4       INSV    R0, #2, #1, FLAGS              ;
                                      0144 C3 9F 001CA       PUSHAB  P.ABS                          ; 0581
                            64             01 FB 001CE       CALLS   #1, CLI$PRESENT                ;
                            06             50 E9 001D1       BLBC    R0, 12$                        ;
                         14 AE        03C0 8F A8 001D4       BISW2   #960, FLAGS+1                  ; 0583
                                        14 AE 95 001DA 12$:  TSTB    FLAGS                          ; 0591
                                           19 19 001DD       BLSS    13$                            ;
                  14     14 AE             06 E0 001DF       BBS     #6, FLAGS, 13$                 ; 0592
                            10          15 AE E8 001E4       BLBS    FLAGS+1, 13$                   ; 0593
                  0B     15 AE             03 E0 001E8       BBS     #3, FLAGS+1, 13$               ; 0594
                  06     15 AE             02 E0 001ED       BBS     #2, FLAGS+1, 13$               ; 0595
                         14 AE        01C0 8F A8 001F2       BISW2   #448, FLAGS                    ; 0597
                  0E     14 AE             02 E1 001F8 13$:  BBC     #2, FLAGS, 14$                 ; 0604
                            07   00000000G 00 E8 001FD       BLBS    CTL$GQ_PROCPRIV, 14$           ; 0607
                            7E        2804 8F 3C 00204       MOVZWL  #10244, -(SP)                  ; 0610
                                           34 11 00209       BRB     15$                            ;
                         18 AE             06 D0 0020B 14$:  MOVL    #6, ARGLST                     ; 0619
                         1C AE          14 AE 9E 0020F       MOVAB   FLAGS, ARGLST+4                ; 0620
                         20 AE             6E D0 00214       MOVL    BLOCK_COUNT, ARGLST+8          ; 0621
                         24 AE          08 AE 7D 00218       MOVQ    BUFFER_COUNT, ARGLST+12        ; 0622
                         2C AE          10 AE D0 0021D       MOVL    EXTEND, ARGLST+20              ; 0624
                         30 AE          04 AE D0 00222       MOVL    NET_BLOCK_COUNT, ARGLST+24     ; 0625
                                        18 AE 9F 00227       PUSHAB  ARGLST                         ; 0627
                                 00000000V EF 9F 0022A       PUSHAB  SETRMSKNL                      ;
                  00000000G 00             02 FB 00230       CALLS   #2, SYS$CMKRNL                 ;
                            52             50 D0 00237       MOVL    R0, STATUS                     ;
                            09             52 E8 0023A       BLBS    STATUS, 16$                    ;
                                           52 DD 0023D       PUSHL   STATUS                         ; 0628
                  00000000G 00             01 FB 0023F 15$:  CALLS   #1, LIB$SIGNAL                 ;
                                              04 00246 16$:  RET                                    ; 0631

; Routine Size:  583 bytes,    Routine Base:  $CODE$ + 016F
```

```
1 ROUTINE setrmsknl (flags, block_count, buffer_count, prolog, extend, net_block_count) =
2 BEGIN
2 !++
2 !
2 ! This is the kernel mode routine that actually sets the RMS defaults
2 !
2 ! Inputs:
2 !       FLAGS - address of flags longword
2 !       BLOCK_COUNT - address of block count
2 !       BUFFER_COUNT - address of buffer count
2 !       PROLOG - address of prologue level
2 !       EXTEND - address of extend quantity
2 !       NET_BLOCK_COUNT - address of network block count
2 !
2 ! Outputs:
2 !       None.  The RMS defaults are reset accordingly.
2 !
2 !--
2
2 MAP flags : REF $BBLOCK;
2
2 !
2 ! See whether the mods are for the system, or simply for this process.
2 !
2 IF .flags[set$v_system]                     ! Make system mods
2 THEN
3     BEGIN
3     IF .flags[set$v_block]                  ! /BLOCK_COUNT
3     THEN
3         sys$gb_dfmbc = .block_count;
3
3     IF .flags[set$v_netblk]                 ! /NETWORK
3     THEN
3         sys$gb_dfnbc = .net_block_count;
3
3     IF .flags[set$v_buffer]                 ! BUFFER_COUNT
3     THEN
4         BEGIN
4         IF .flags[set$v_disk]               ! /DISK
4         THEN sys$gb_dfmbfsdk = .buffer_count;
4         IF .flags[set$v_tape]               ! /MAGTAPE
4         THEN sys$gb_dfmbfsmt = .buffer_count;
4         IF .flags[set$v_unit]               ! /UNIT_RECORD
4         THEN sys$gb_dfmbfsur = .buffer_count;
4         IF .flags[set$v_hash]               ! /HASH
4         THEN sys$gb_dfmbfhsh = .buffer_count;
4         IF .flags[set$v_index]              ! /INDEXED
4         THEN sys$gb_dfmbfidx = .buffer_count;
4         IF .flags[set$v_rel]                ! /RELATIVE
4         THEN sys$gb_dfmbfrel = .buffer_count;
3         END;
3     IF .flags[set$v_prolog]                 ! /PROLOG
3     THEN sys$gb_rmsprolog = .prolog;
3     IF .flags[set$v_extend]                 ! /EXTEND
3     THEN sys$gw_rmsextend = .extend;
3     END
3
```

```
;  697        0689  3 !
;  698        0690  3 ! If not /SYSTEM, then it must be for the process.
;  699        0691  3 !
;  700        0692  2 ELSE
;  701        0693  3     BEGIN                                 ! Make process mods
;  702        0694  3     IF .flags[set$v_block]                ! /BLOCK_COUNT
;  703        0695  3     THEN
;  704        0696  3         pio$gb_dfmbc = .block_count;
;  705        0697  3     IF .flags[set$v_netblk]               ! /NETWORK
;  706        0698  3     THEN
;  707        0699  3         pio$gb_dfnbc = .net_block_count;
;  708        0700  3     IF .flags[set$v_buffer]               ! /BUFFER_COUNT
;  709        0701  3     THEN
;  710        0702  4         BEGIN
;  711        0703  4         IF .flags[set$v_disk]             ! /DISK
;  712        0704  4         THEN pio$gb_dfmbfsdk = .buffer_count;
;  713        0705  4         IF .flags[set$v_tape]             ! /MAGTAPE
;  714        0706  4         THEN pio$gb_dfmbfsmt = .buffer_count;
;  715        0707  4         IF .flags[set$v_unit]             ! /UNIT_RECORD
;  716        0708  4         THEN pio$gb_dfmbfsur = .buffer_count;
;  717        0709  4         IF .flags[set$v_hash]             ! /HASHED
;  718        0710  4         THEN pio$gb_dfmbfhsh = .buffer_count;
;  719        0711  4         IF .flags[set$v_index]            ! /INDEXED
;  720        0712  4         THEN pio$gb_dfmbfidx = .buffer_count;
;  721        0713  4         IF .flags[set$v_rel]              ! /RELATIVE
;  722        0714  4         THEN pio$gb_dfmbfrel = .buffer_count;
;  723        0715  3         END;
;  724        0716  3     IF .flags[set$v_prolog]               ! /PROLOG
;  725        0717  3     THEN pio$gb_rmsprolog = .prolog;
;  726        0718  3     IF .flags[set$v_extend]               ! /EXTEND
;  727        0719  3     THEN pio$gw_rmsextend = .extend;
;  728        0720  2     END;
;  729        0721  2
;  730        0722  2 RETURN 1;
;  731        0723  1 END;


                                          0000 00000 SETRMSKNL:
                                                          .WORD    Save nothing                         ; 0632
                 50        04   AC   D0 00002             MOVL     FLAGS, R0                            ; 0656
  7E             60             02   E1 00006             BBC      #2, (R0), 10$                        ;
  08             60             03   E1 0000A             BBC      #3, (R0), 1$                         ; 0659
     00000000G   00        08   AC   90 0000E             MOVB     BLOCK_COUNT, SYS$GB_DFMBC            ; 0661
  08             60             0E   E1 00016 1$:         BBC      #14, (R0), 2$                        ; 0663
     00000000G   00        18   AC   90 0001A             MOVB     NET_BLOCK_COUNT, SYS$GB_DFNBC        ; 0665
  48             60             04   E1 00022 2$:         BBC      #4, (R0), 8$                         ; 0667
  08             60             06   E1 00026             BBC      #6, (R0), 3$                         ; 0670
     00000000G   00        0C   AC   90 0002A             MOVB     BUFFER_COUNT, SYS$GB_DFMBFSDK        ; 0671
                                60   95 00032 3$:         TSTB     (R0)                                 ; 0672
                                08   18 00034             BGEQ     4$                                   ;
     00000000G   00        0C   AC   90 00036             MOVB     BUFFER_COUNT, SYS$GB_DFMBFSMT        ; 0673
                 08        01   A0   E9 0003E 4$:         BLBC     1(R0), 5$                            ; 0674
     00000000G   00        0C   AC   90 00042             MOVB     BUFFER_COUNT, SYS$GB_DFMBFSUR        ; 0675
  08             60             0C   E1 0004A 5$:         BBC      #12, (R0), 6$                        ; 0676
```

```
         00000000G  00       0C   AC   90 0004E          MOVB    BUFFER_COUNT, SYS$GB_DFMBFHSH          ; 0677
    08              60             0B   E1 00056 6$:      BBC     #11, (R0), 7$                          ; 0678
         00000000G  00       0C   AC   90 0005A          MOVB    BUFFER_COUNT, SYS$GB_DFMBFIDX          ; 0679
    08              60             0A   E1 00062 7$:      BBC     #10, (R0), 8$                          ; 0680
         00000000G  00       0C   AC   90 00066          MOVB    BUFFER_COUNT, SYS$GB_DFMBFREL          ; 0681
    08              60             05   E1 0006E 8$:      BBC     #5, (R0), 9$                           ; 0683
         00000000G  00       10   AC   90 00072          MOVB    PROLOG, SYS$GB_RMSPROLOG               ; 0684
    7A              60             0D   E1 0007A 9$:      BBC     #13, (R0), 19$                         ; 0685
         00000000G  00       14   AC   B0 0007E          MOVW    EXTEND, SYS$GW_RMSEXTEND               ; 0686
                                   7C   11 00086          BRB     20$                                    ; 0656
    08              60             03   E1 00088 10$:     BBC     #3, (R0), 11$                          ; 0694
         00000000G  00       08   AC   90 0008C          MOVB    BLOCK_COUNT, PIO$GB_DFMBC              ; 0696
    08              60             0E   E1 00094 11$:     BBC     #14, (R0), 12$                         ; 0697
         00000000G  00       18   AC   90 00098          MOVB    NET_BLOCK_COUNT, PIO$GB_DFNBC          ; 0699
    48              60             04   E1 000A0 12$:     BBC     #4, (R0), 18$                          ; 0700
    08              60             06   E1 000A4          BBC     #6, (R0), 13$                          ; 0703
         00000000G  00       0C   AC   90 000A8          MOVB    BUFFER_COUNT, PIO$GB_DFMBFSDK          ; 0704
                                   60   95 000B0 13$:     TSTB    (R0)                                   ; 0705
                                   08   18 000B2          BGEQ    14$                                    ;
         00000000G  00       0C   AC   90 000B4          MOVB    BUFFER_COUNT, PIO$GB_DFMBFSMT          ; 0706
                    08       01   A0   E9 000BC 14$:     BLBC    1(R0), 15$                             ; 0707
         00000000G  00       0C   AC   90 000C0          MOVB    BUFFER_COUNT, PIO$GB_DFMBFSUR          ; 0708
    08              60             0C   E1 000C8 15$:     BBC     #12, (R0), 16$                         ; 0709
         00000000G  00       0C   AC   90 000CC          MOVB    BUFFER_COUNT, PIO$GB_DFMBFHSH          ; 0710
    08              60             0B   E1 000D4 16$:     BBC     #11, (R0), 17$                         ; 0711
         00000000G  00       0C   AC   90 000D8          MOVB    BUFFER_COUNT, PIO$GB_DFMBFIDX          ; 0712
    08              60             0A   E1 000E0 17$:     BBC     #10, (R0), 18$                         ; 0713
         00000000G  00       0C   AC   90 000E4          MOVB    BUFFER_COUNT, PIO$GB_DFMBFREL          ; 0714
    08              60             05   E1 000EC 18$:     BBC     #5, (R0), 19$                          ; 0716
         00000000G  00       10   AC   90 000F0          MOVB    PROLOG, PIO$GB_RMSPROLOG               ; 0717
    08              60             0D   E1 000F8 19$:     BBC     #13, (R0), 20$                         ; 0718
         00000000G  00       14   AC   B0 000FC          MOVW    EXTEND, PIO$GW_RMSEXTEND               ; 0719
                    50             01   D0 00104 20$:     MOVL    #1, R0                                 ; 0722
                                        04 00107          RET                                            ; 0723

; Routine Size:  264 bytes,    Routine Base:  $CODE$ + 03B6
```

```
1 GLOBAL ROUTINE set$working_set : NOVALUE =
2 BEGIN
2 !++
2 !
2 ! This routine implements the SET WORKING_SSET command.  The values and
2 ! qualifiers are collected and checked, then a kernel call is made to
2 ! actually set the parameters.
2 !
2 ! Inputs:
2 !       None.  The CLI is interrogated.
2 !
2 ! Outputs:
2 !       None.  The working set defaults are changed.
2 !
2 !--
2
2 LOCAL
2     status,                                       ! Status return
2     limit,                                        ! Working set limit
2     quota,                                        ! Working set quota
2     extent,                                       ! Working set extent
2     specified_limit,                              ! And the real values that
2     specified_quota,                              ! were specified by the
2     specified_extent,                             ! user before juggling
2     min_wset,                                     ! Minimum guaranteed working set
2     auth_limit,                                   ! Authorized limit
2     auth_extent,                                  ! Authorized extent
2     flags : $BBLOCK[4] INITIAL(0),                ! Flags longword
2     desc : $BBLOCK[dsc$c_s_bln],                  ! General descriptor
2     arglist : VECTOR[5];                          ! Argument list for kernel call
2
2 BIND
2     phd = .ctl$gl_phd : $BBLOCK;                  ! Point to this process's PHD
2
2 !
2 ! Initialize the descriptor, and calculate some quantities that are handy to
2 ! have.  These are the authorized working set limit, the minimum working set,
2 ! and the authorized extend limit.
2 !
2 $init_dyndesc(desc);                              ! Make the descriptor dynamic
2 auth_limit = .phd[phd$w_wsauth] - .phd[phd$w_wslist] + 1;
2 auth_extent = .phd[phd$w_wsauthext] - .phd[phd$w_wslist] + 1;
2 min_wset = .phd[phd$w_wsdyn] - .phd[phd$w_wslist] + 2*.phd[phd$w_wsfluid] + 3;
2
2 !
2 ! Get the /[NO]ADJUST and /[NO]LOG flags.
2 !
2 ! If the /ADJUST qualifier is present explicitly, then set that flag, and
2 ! in the process note whether it was /ADJUST or /NOADJUST.
2 !
2 status = flags[set$v_adjust]                      ! Get the /ADJ or /NOADJ
2       = cli$present(%ASCID 'ADJUST');             ! but only use it if
2 flags[set$v_expadj] = (.status NEQ cli$_absent);! explicitly specified.
2
2 status = flags[set$v_log]                         ! Same for /LOG
2       = cli$present(%ASCID 'LOG');
2 flags[set$v_explog] = (.status NEQ cli$_absent);
```

```
2
2
2 !
2 ! If a new limit is given, then check that the value is valid, and
2 ! then apply some common sense bounds checking.  If no new limit was set,
2 ! compute the current one.
2 !
3 IF (flags[set$v_limit] = cli$get_value(%ASCID 'LIMIT', desc))
2 THEN
3     BEGIN                                         ! Convert from ASCII to number
3     IF NOT lib$cvt_dtb(.desc[dsc$w_length],
3                       .desc[dsc$a_pointer],
3                       specified_limit)
3     THEN                                          ! If an error, signal it
4         BEGIN
4         SIGNAL(set$_invquaval, 2, desc, %ASCID 'LIMIT');
4         RETURN;
4         END
3     ELSE                                          ! If the value is good, check
4         BEGIN                                     ! that it is within reasonable
4         LOCAL temp;                               ! bounds.
4         temp = MAX(.min_wset, .specified_limit);! No lower than the minimum,
4         limit = MIN(.temp, .auth_limit);          ! No higher than the authorized
3         END;
3     END
3 !
3 ! If no new limit was given, compute the current one.
3 !
2 ELSE limit = specified_limit
2           = .phd[phd$w_dfwscnt] - .phd[phd$w_wslist] + 1;
2
2
2
2 !
2 ! If a new value given, validate it and make some common sense
2 ! range checks
2 !
3 IF (flags[set$v_quota] = cli$get_value(%ASCID 'QUOTA', desc))
2 THEN
3     BEGIN                                         ! Convert from ASCII to number
3     IF NOT lib$cvt_dtb(.desc[dsc$w_length],
3                       .desc[dsc$a_pointer],
3                       specified_quota)
3     THEN                                          ! If an error, signal it
4         BEGIN
4         SIGNAL(set$_invquaval, 2, desc, %ASCID 'QUOTA');
4         RETURN;
4         END
3     ELSE                                          ! Otherwise make some
4         BEGIN                                     ! bounds checks
4         LOCAL temp;
4         temp = MAX(.min_wset, .specified_quota);! No lower than the minimum,
4         quota = MIN(.temp, .auth_limit);          ! No higher than the authorized
3         END;
3     END
3 !
3 ! If no new quota given, compute the current one.
```

```
3 !
2 ELSE quota = specified_quota
2           = .phd[phd$w_wsquota] - .phd[phd$w_wslist] + 1;
2
2
2 !
2 ! If a new extent is given, validate and make the usual checks.
2 !
3 IF (flags[set$v_extent] = cli$get_value(%ASCID 'EXTENT', desc))
2 THEN
3     BEGIN                                       ! Convert from ASCII to a number
3     IF NOT lib$cvt_dtb(.desc[dsc$w_length],
3                        .desc[dsc$a_pointer],
3                        specified_extent)
3     THEN                                        ! If an error, signal it.
4         BEGIN
4         SIGNAL(set$_invquaval, 2, desc, %ASCID 'EXTENT');
4         RETURN;
4         END
3     ELSE
4         BEGIN                                   ! Make some bounds checks
4         LOCAL temp;
4         temp = MAX(.min_wset, .specified_extent);! No lower than the minimum,
4         extent = MIN(.temp, .auth_extent);      ! No higher than the authorized
3         END;
3     END
3 !
3 ! If no new extent given, compute the current one.
3 !
2 ELSE extent = specified_extent
2           = .phd[phd$w_wsextent] - .phd[phd$w_wslist] + 1;
2
2 !
2 ! Now for some further consistency checking.  The general rule is that
2 !
2 !               LIMIT < QUOTA < EXTENT
2 !
2 ! Because LIMIT is what the working set is at image rundown,
2 !         QUOTA is what a process is guaranteed it can grow to, and
2 !         EXTENT is what it might grow to if there's extra memory around.
2 ! In addition, the relative importance of the qualifiers is that EXTENT is
2 ! relatively more important than QUOTA, which is more important than LIMIT.
2 ! These are the general rules that govern the mess that follows.
2 !
2 ! If all the EXTENT, QUOTA, and LIMIT were changed, or the EXTENT and QUOTA,
2 ! or just the EXTENT, the EXTENT is taken as the most important, and the
2 ! other two values get adjusted accordingly.
2 !
3 IF (.flags[set$v_extent] AND .flags[set$v_quota])
3 OR (.flags[set$v_extent] AND NOT (.flags[set$v_quota] OR .flags[set$v_limit]))
2 THEN
3     BEGIN
3     quota = MIN(.extent, .quota);        ! QUOTA < EXTENT
3     limit = MIN(.quota, .limit);         ! and LIMIT < QUOTA
3     END
```

```
  3 !
  3 ! If LIMIT and QUOTA were set only, or just QUOTA, then reset EXTENT and
  3 ! juggle with the LIMIT.
  3 !
  2 ELSE IF .flags[set$v_quota]
  2 THEN
  3     BEGIN
  3     extent = MAX(.quota, .extent);         !  QUOTA < EXTENT
  3     limit = MIN(.quota, .limit);           ! and LIMIT < QUOTA
  3     END
  3 !
  3 ! If LIMIT and EXTENT only, then reset LIMIT, then juggle QUOTA.
  3 !
  3 ELSE IF (.flags[set$v_limit] AND .flags[set$v_extent])
  2 THEN
  3     BEGIN
  3     limit = MIN(.extent, .limit);          ! Set LIMIT < EXTENT
  3     quota = MAX(.limit, .quota);           ! LIMIT < QUOTA
  3     quota = MIN(.extent, .quota);          ! QUOTA < EXTENT
  3     END
  3 !
  3 ! Finally, if only LIMIT was set, make sure that EXTENT is larger,
  3 ! and that QUOTA is larger.
  3 !
  2 ELSE IF .flags[set$v_limit]
  2 THEN
  3     BEGIN
  3     extent = MAX(.limit, .extent);         ! LIMIT < EXTENT
  3     quota = MAX(.limit, .quota);           ! LIMIT < QUOTA
  2     END;
  2
  2 !
  2 ! Call the kernel-mode routine that actually sets the parameters.
  2 !
  2 arglist[0] = 4;
  2 arglist[1] = .limit;
  2 arglist[2] = .quota;
  2 arglist[3] = .extent;
  2 arglist[4] = flags;
P 3 IF NOT (status = $CMKRNL(ROUTIN = setwrkknl,
  3                          ARGLST = arglist))
  2 THEN
  3     BEGIN
  3     SIGNAL(.status);
  3     RETURN;
  2     END;
  2
  2 !
  2 ! Now for how much to tell the user.  If something was changed, and /NOLOG
  2 ! wasn't specified, then signal the new values.  Also, if /LOG was specified,
  2 ! signal the new values.
  2 !
  3 IF (.flags[set$v_explog] AND .flags[set$v_log]) ! If user specified /LOG
  4 OR ((.specified_limit NEQ .limit OR             ! or if any of the values
  4     .specified_quota NEQ .quota OR              ! were juggled,
  3     .specified_extent NEQ .extent) AND NOT      ! and the user didn't say /NOLOG
  3     (.flags[set$v_explog] AND NOT .flags[set$v_log]))
```

```
                                                        B 13
SETMISC                                                 16-Sep-1984 00:43:54    VAX-11 Bliss-32 V4.0-742            Page  30
V04-000                                                 14-Sep-1984 12:09:11    [CLIUTL.SRC]SETMISC.B32;1               (10)

;  961          0952  2 THEN SIGNAL(set$_newlims, 3,                        ! signal an informational
;  962          0953  2             .limit,
;  963          0954  2             .quota,
;  964          0955  2             .extent);
;  965          0956  2
;  966          0957  2 RETURN 1;
;  967          0958  1 END;


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                      00 00 54 53 55 4A 44 41  001B4 P.ABV:  .ASCII  \ADJUST\<0><0>                                      :
                                   010E0006  001BC P.ABU:  .LONG   17694726                                            :
                                   00000000' 001C0         .ADDRESS P.ABV                                              :
                                  00 47 4F 4C  001C4 P.ABX:  .ASCII  \LOG\<0>                                            :
                                   010E0003  001C8 P.ABW:  .LONG   17694723                                            :
                                   00000000' 001CC         .ADDRESS P.ABX                                              :
                      00 00 00 54 49 4D 49 4C  001D0 P.ABZ:  .ASCII  \LIMIT\<0><0><0>                                    :
                                   010E0005  001D8 P.ABY:  .LONG   17694725                                            :
                                   00000000' 001DC         .ADDRESS P.ABZ                                              :
                      00 00 00 54 49 4D 49 4C  001E0 P.ACB:  .ASCII  \LIMIT\<0><0><0>                                    :
                                   010E0005  001E8 P.ACA:  .LONG   17694725                                            :
                                   00000000' 001EC         .ADDRESS P.ACB                                              :
                      00 00 00 41 54 4F 55 51  001F0 P.ACD:  .ASCII  \QUOTA\<0><0><0>                                    :
                                   010E0005  001F8 P.ACC:  .LONG   17694725                                            :
                                   00000000' 001FC         .ADDRESS P.ACD                                              :
                      00 00 00 41 54 4F 55 51  00200 P.ACF:  .ASCII  \QUOTA\<0><0><0>                                    :
                                   010E0005  00208 P.ACE:  .LONG   17694725                                            :
                                   00000000' 0020C         .ADDRESS P.ACF                                              :
                      00 00 54 4E 45 54 58 45  00210 P.ACH:  .ASCII  \EXTENT\<0><0>                                      :
                                   010E0006  00218 P.ACG:  .LONG   17694726                                            :
                                   00000000' 0021C         .ADDRESS P.ACH                                              :
                      00 00 54 4E 45 54 58 45  00220 P.ACJ:  .ASCII  \EXTENT\<0><0>                                      :
                                   010E0006  00228 P.ACI:  .LONG   17694726                                            :
                                   00000000' 0022C         .ADDRESS P.ACJ                                              :


                                                                    .PSECT  $CODE$,NOWRT,2

                                       0FFC 00000          .ENTRY  SET$WORKING_SET, Save R2,R3,R4,R5,R6,R7,R8,-; 0724
                                                                   R9,R10,R11                                          :
                      5B 00000000G  00  9E 00002          MOVAB   LIB$CVT_DTB, R11                                    :
                      5A 00000000G  00  9E 00009          MOVAB   CLI$GET_VALUE, R10                                  :
                      59 00000000'  EF  9E 00010          MOVAB   P.ABU, R9                                           :
                      5E            2C  C2 00017          SUBL2   #44, SP                                             :
                               0C   AE  D4 0001A          CLRL    FLAGS                                               ; 0725
                      56 00000000G  00  D0 0001D          MOVL    CTL$GL_PHD, R6                                      ; 0756
                   24 AE 020E0000   8F  D0 00024          MOVL    #34471936, DESC                                     ; 0763
                               28   AE  D4 0002C          CLRL    DESC+4                                              :
                      57       08   A6  3C 0002F          MOVZWL  8(R6), R7                                           ; 0764
                      55       0A   A6  3C 00033          MOVZWL  10(R6), R5                                          :
                      55            57  C2 00037          SUBL2   R7, R5                                              :
                      52       01   A5  9E 0003A          MOVAB   1(R5), AUTH_LIMIT                                   :
                      51       14   A6  3C 0003E          MOVZWL  20(R6), R1                                          ; 0765
                      51            57  C2 00042          SUBL2   R7, R1                                              :
```

```
                                   55      01  A1  9E 00045        MOVAB   1(R1), AUTH_EXTENT                  ;
                                   51      0E  A6  3C 00049        MOVZWL  14(R6), R1                          ; 0766
                                   51          57  C2 0004D        SUBL2   R7, R1                              ;
                                   50      74  A6  3C 00050        MOVZWL  116(R6), R0                         ;
                                   53    03 A140   3E 00054        MOVAW   3(R1)[R0], MIN_WSET                 ;
                                               59  DD 00059        PUSHL   R9                                  ; 0775
                        00000000G  00          01  FB 0005B        CALLS   #1, CLI$PRESENT                     ;
   0C    AE          01            06          50  F0 00062        INSV    R0, #6, #1, FLAGS                   ;
                                   58          50  D0 00068        MOVL    R0, STATUS                          ;
                                               50  D4 0006B        CLRL    R0                                  ; 0776
                        00000000G  8F          58  D1 0006D        CMPL    STATUS, #CLI$_ABSENT                ;
                                               02  13 00074        BEQL    1$                                  ;
                                               50  D6 00076        INCL    R0                                  ;
   0C    AE          01            05          50  F0 00078 1$:    INSV    R0, #5, #1, FLAGS                   ;
                                           0C  A9  9F 0007E        PUSHAB  P.ABW                               ; 0779
                        00000000G  00          01  FB 00081        CALLS   #1, CLI$PRESENT                     ;
   0C    AE          01            00          50  F0 00088        INSV    R0, #0, #1, FLAGS                   ;
                                   58          50  D0 0008E        MOVL    R0, STATUS                          ;
                                               50  D4 00091        CLRL    R0                                  ; 0780
                        00000000G  8F          58  D1 00093        CMPL    STATUS, #CLI$_ABSENT                ;
                                               02  13 0009A        BEQL    2$                                  ;
                                               50  D6 0009C        INCL    R0                                  ;
   0C    AE          01            01          50  F0 0009E 2$:    INSV    R0, #1, #1, FLAGS                   ;
                                           24  AE  9F 000A4        PUSHAB  DESC                                ; 0788
                                           1C  A9  9F 000A7        PUSHAB  P.ABY                               ;
                                   6A          02  FB 000AA        CALLS   #2, CLI$GET_VALUE                   ;
   0C    AE          01            02          50  F0 000AD        INSV    R0, #2, #1, FLAGS                   ;
                                   2C          50  E9 000B3        BLBC    R0, 6$                              ;
                                               5E  DD 000B6        PUSHL   SP                                  ; 0791
                                           2C  AE  DD 000B8        PUSHL   DESC+4                              ; 0792
                                   7E      2C  AE  3C 000BB        MOVZWL  DESC, -(SP)                         ; 0791
                                   6B          03  FB 000BF        CALLS   #3, LIB$CVT_DTB                     ;
                                   05          50  E8 000C2        BLBS    R0, 3$                              ;
                                           2C  A9  9F 000C5        PUSHAB  P.ACA                               ; 0796
                                               4C  11 000C8        BRB     8$                                  ;
                                   50          53  D0 000CA 3$:    MOVL    MIN_WSET, R0                        ; 0802
                                   6E          50  D1 000CD        CMPL    R0, SPECIFIED_LIMIT                 ;
                                               03  18 000D0        BGEQ    4$                                  ;
                                   50          6E  D0 000D2        MOVL    SPECIFIED_LIMIT, R0                 ;
                                   52          50  D1 000D5 4$:    CMPL    R0, AUTH_LIMIT                      ; 0803
                                               03  15 000D8        BLEQ    5$                                  ;
                                   50          52  D0 000DA        MOVL    AUTH_LIMIT, R0                      ;
                                   54          50  D0 000DD 5$:    MOVL    R0, LIMIT                           ;
                                               0F  11 000E0        BRB     7$                                  ; 0788
                                   51      1A  A6  3C 000E2 6$:    MOVZWL  26(R6), R1                          ; 0810
                                   51          57  C2 000E6        SUBL2   R7, R1                              ;
                                               51  D6 000E9        INCL    R1                                  ;
                                   6E          51  D0 000EB        MOVL    R1, SPECIFIED_LIMIT                 ;
                                   54          51  D0 000EE        MOVL    R1, LIMIT                           ;
                                           24  AE  9F 000F1 7$:    PUSHAB  DESC                                ; 0818
                                           3C  A9  9F 000F4        PUSHAB  P.ACC                               ;
                                   6A          02  FB 000F7        CALLS   #2, CLI$GET_VALUE                   ;
   0C    AE          01            03          50  F0 000FA        INSV    R0, #3, #1, FLAGS                   ;
                                   2F          50  E9 00100        BLBC    R0, 12$                             ;
                                           04  AE  9F 00103        PUSHAB  SPECIFIED_QUOTA                     ; 0821
                                           2C  AE  DD 00106        PUSHL   DESC+4                              ; 0822
                                   7E      2C  AE  3C 00109        MOVZWL  DESC, -(SP)                         ; 0821
```

```
                                                       D 13
SETMISC                                                16-Sep-1984 00:43:54    VAX-11 Bliss-32 V4.0-742                 Page  32
V04-000                                                14-Sep-1984 12:09:11    [CLIUTL.SRC]SETMISC.B32;1                     (10)

                                        6B            03 FB 0010D         CALLS    #3, LIB$CVT_DTB                 ;
                                        05            50 E8 00110         BLBS     R0, 9$                          ;
                                                4C    A9 9F 00113         PUSHAB   P.ACE                           ; 0826
                                                      4C 11 00116 8$:     BRB      14$                             ;
                                        50            53 D0 00118 9$:     MOVL     MIN_WSET, R0                    ; 0832
                                04      AE            50 D1 0011B         CMPL     R0, SPECIFIED_QUOTA             ;
                                                      04 18 0011F         BGEQ     10$                             ;
                                        50      04    AE D0 00121         MOVL     SPECIFIED_QUOTA, R0             ;
                                        52            50 D1 00125 10$:    CMPL     R0, AUTH_LIMIT                  ; 0833
                                                      03 15 00128         BLEQ     11$                             ;
                                        50            52 D0 0012A         MOVL     AUTH_LIMIT, R0                  ;
                                        52            50 D0 0012D 11$:    MOVL     R0, QUOTA                       ;
                                                      0D 11 00130         BRB      13$                             ; 0818
                                        52      18    A6 3C 00132 12$:    MOVZWL   24(R6), R2                      ; 0840
                                        52            57 C2 00136         SUBL2    R7, R2                          ;
                                                      52 D6 00139         INCL     R2                              ;
                                04      AE            52 D0 0013B         MOVL     R2, SPECIFIED_QUOTA             ;
                                                24    AE 9F 0013F 13$:    PUSHAB   DESC                            ; 0847
                                                5C    A9 9F 00142         PUSHAB   P.ACG                           ;
                                        6A            02 FB 00145         CALLS    #2, CLI$GET_VALUE               ;
    0C    AE          01                04            50 F0 00148         INSV     R0, #4, #1, FLAGS               ;
                                        40            50 E9 0014E         BLBC     R0, 18$                         ;
                                                08    AE 9F 00151         PUSHAB   SPECIFIED_EXTENT                ; 0850
                                                2C    AE DD 00154         PUSHL    DESC+4                          ; 0851
                                        7E      2C    AE 3C 00157         MOVZWL   DESC, -(SP)                     ; 0850
                                        6B            03 FB 0015B         CALLS    #3, LIB$CVT_DTB                 ;
                                        16            50 E8 0015E         BLBS     R0, 15$                         ;
                                                6C    A9 9F 00161         PUSHAB   P.ACI                           ; 0855
                                                28    AE 9F 00164 14$:    PUSHAB   DESC                            ;
                                                      02 DD 00167         PUSHL    #2                              ;
                                          0077132A    8F DD 00169         PUSHL    #7803690                        ;
                       00000000G        00            04 FB 0016F         CALLS    #4, LIB$SIGNAL                  ;
                                                         04 00176         RET                                      ; 0854
                                08      AE            53 D1 00177 15$:    CMPL     R3, SPECIFIED_EXTENT            ; 0861
                                                      04 18 0017B         BGEQ     16$                             ;
                                        53      08    AE D0 0017D         MOVL     SPECIFIED_EXTENT, R3            ;
                                        50            53 D0 00181 16$:    MOVL     R3, TEMP                        ;
                                        55            50 D1 00184         CMPL     R0, AUTH_EXTENT                 ; 0862
                                                      03 15 00187         BLEQ     17$                             ;
                                        50            55 D0 00189         MOVL     AUTH_EXTENT, R0                 ;
                                        53            50 D0 0018C 17$:    MOVL     R0, EXTENT                      ;
                                                      10 11 0018F         BRB      19$                             ; 0847
                                        56      16    A6 3C 00191 18$:    MOVZWL   22(R6), R6                      ; 0869
                                        56            57 C2 00195         SUBL2    R7, R6                          ;
                                                      56 D6 00198         INCL     R6                              ;
                                08      AE            56 D0 0019A         MOVL     R6, SPECIFIED_EXTENT            ;
                                        53            56 D0 0019E         MOVL     R6, EXTENT                      ;
                      29        0C      AE            04 E1 001A1 19$:    BBC      #4, FLAGS, 22$                  ; 0888
                      0F        0C      AE            03 E0 001A6         BBS      #3, FLAGS, 20$                  ;
                      1F        0C      AE            04 E1 001AB         BBC      #4, FLAGS, 22$                  ; 0889
                      1F        0C      AE            03 E0 001B0         BBS      #3, FLAGS, 23$                  ;
                      15        0C      AE            02 E0 001B5         BBS      #2, FLAGS, 22$                  ;
                                        50            53 D0 001BA 20$:    MOVL     EXTENT, R0                      ; 0892
                                        52            50 D1 001BD         CMPL     R0, QUOTA                       ;
                                                      03 15 001C0         BLEQ     21$                             ;
                                        50            52 D0 001C2         MOVL     QUOTA, R0                       ;
                                        52            50 D0 001C5 21$:    MOVL     R0, QUOTA                       ;
```

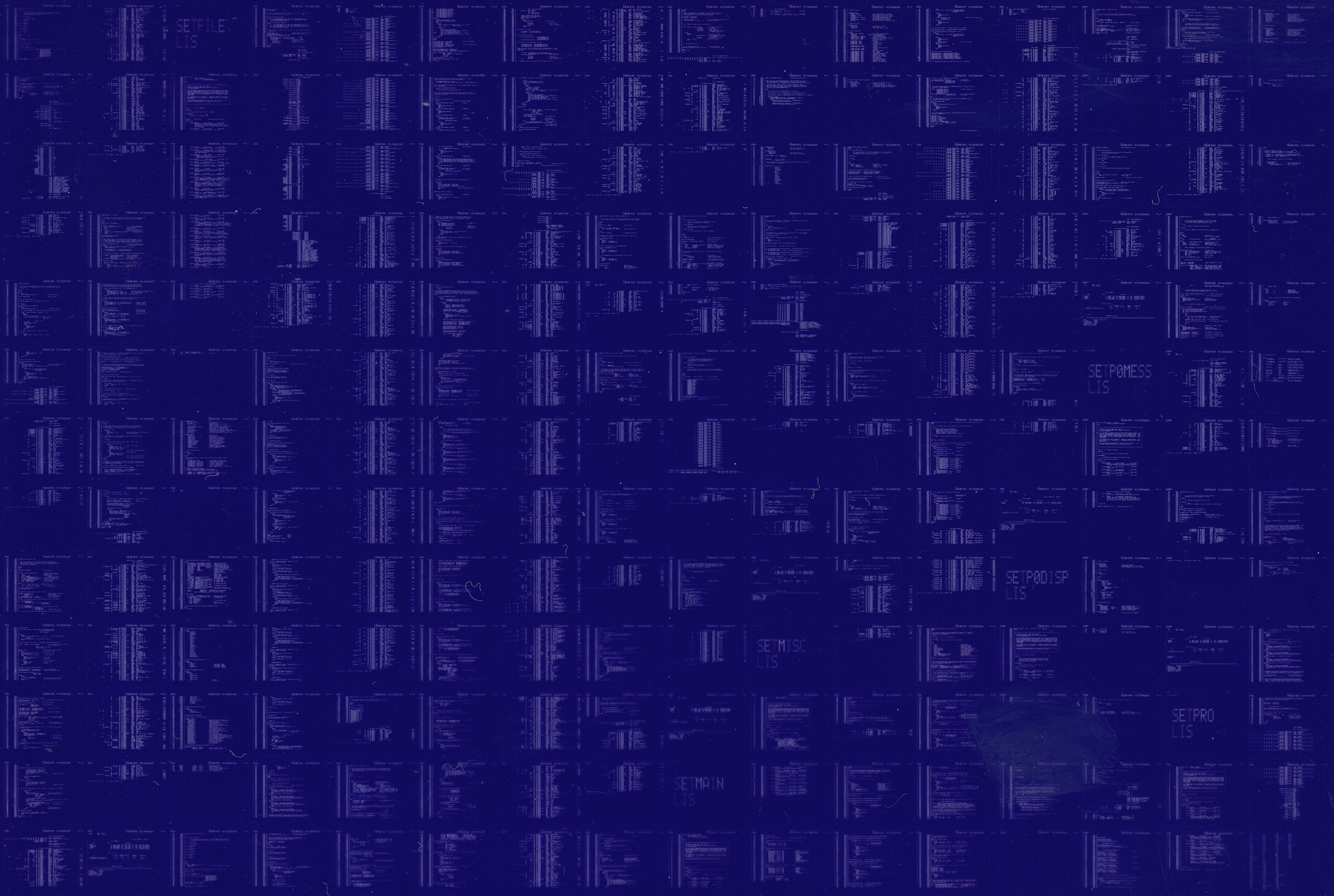
```
                        54                50  D1 001C8         CMPL    R0, LIMIT                                  ; 0893
                                          1D  14 001CB         BGTR    25$                                        ;
                                          1E  11 001CD         BRB     26$                                        ;
        1E      0C      AE                03  E1 001CF  22$:   BBC     #3, FLAGS, 27$                             ; 0899
                        50                52  D0 001D4  23$:   MOVL    QUOTA, R0                                  ; 0902
                        53                50  D1 001D7         CMPL    R0, EXTENT                                 ;
                                          03  18 001DA         BGEQ    24$                                        ;
                        50                53  D0 001DC         MOVL    EXTENT, R0                                 ;
                        53                50  D0 001DF  24$:   MOVL    R0, EXTENT                                 ;
                        50                52  D0 001E2         MOVL    QUOTA, R0                                  ; 0903
                        54                50  D1 001E5         CMPL    R0, LIMIT                                  ;
                                          03  15 001E8         BLEQ    26$                                        ;
                        50                54  D0 001EA  25$:   MOVL    LIMIT, R0                                  ;
                        54                50  D0 001ED  26$:   MOVL    R0, LIMIT                                  ;
                                          4E  11 001F0         BRB     34$                                        ; 0899
        49      0C      AE                02  E1 001F2  27$:   BBC     #2, FLAGS, 34$                             ; 0908
        23      0C      AE                04  E1 001F7         BBC     #4, FLAGS, 30$                             ;
                        50                53  D0 001FC         MOVL    EXTENT, R0                                 ; 0911
                        54                50  D1 001FF         CMPL    R0, LIMIT                                  ;
                                          03  15 00202         BLEQ    28$                                        ;
                        50                54  D0 00204         MOVL    LIMIT, R0                                  ;
                        54                50  D0 00207  28$:   MOVL    R0, LIMIT                                  ;
                        52                50  D1 0020A         CMPL    R0, QUOTA                                  ; 0912
                                          03  18 0020D         BGEQ    29$                                        ;
                        50                52  D0 0020F         MOVL    QUOTA, R0                                  ;
                        52                50  D0 00212  29$:   MOVL    R0, QUOTA                                  ;
                        50                53  D0 00215         MOVL    EXTENT, R0                                 ; 0913
                        52                50  D1 00218         CMPL    R0, QUOTA                                  ;
                                          1D  14 0021B         BGTR    32$                                        ;
                                          1E  11 0021D         BRB     33$                                        ;
        1C      0C      AE                02  E1 0021F  30$:   BBC     #2, FLAGS, 34$                             ; 0919
                        50                54  D0 00224         MOVL    LIMIT, R0                                  ; 0922
                        53                50  D1 00227         CMPL    R0, EXTENT                                 ;
                                          03  18 0022A         BGEQ    31$                                        ;
                        50                53  D0 0022C         MOVL    EXTENT, R0                                 ;
                        53                50  D0 0022F  31$:   MOVL    R0, EXTENT                                 ;
                        50                54  D0 00232         MOVL    LIMIT, R0                                  ; 0923
                        52                50  D1 00235         CMPL    R0, QUOTA                                  ;
                                          03  18 00238         BGEQ    33$                                        ;
                        50                52  D0 0023A  32$:   MOVL    QUOTA, R0                                  ;
                        52                50  D0 0023D  33$:   MOVL    R0, QUOTA                                  ;
                10      AE                04  D0 00240  34$:   MOVL    #4, ARGLIST                                ; 0929
                14      AE                54  D0 00244         MOVL    LIMIT, ARGLIST+4                           ; 0930
                18      AE                52  7D 00248         MOVQ    QUOTA, ARGLIST+8                           ; 0931
                20      AE        0C      AE  9E 0024C         MOVAB   FLAGS, ARGLIST+16                          ; 0933
                                  10      AE  9F 00251         PUSHAB  ARGLIST                                    ; 0935
                           00000000V      EF  9F 00254         PUSHAB  SETWRKKNL                                  ;
           00000000G    00                02  FB 0025A         CALLS   #2, SYS$CMKRNL                             ;
                        58                50  D0 00261         MOVL    R0, STATUS                                 ;
                        0A                58  E8 00264         BLBS    STATUS, 35$                                ;
                                          58  DD 00267         PUSHL   STATUS                                     ; 0938
           00000000G    00                01  FB 00269         CALLS   #1, LIB$SIGNAL                             ;
                                              04 00270         RET                                                ; 0937
        04      0C      AE                01  E1 00271  35$:   BBC     #1, FLAGS, 36$                             ; 0947
                        1A        0C      AE  E8 00276         BLBS    FLAGS, 38$                                 ;
                        54                6E  D1 0027A  36$:   CMPL    SPECIFIED_LIMIT, LIMIT                     ; 0948
                                          0C  12 0027D         BNEQ    37$                                        ;
```

```
                   52        04   AE   D1 0027F          CMPL    SPECIFIED_QUOTA, QUOTA                        ; 0949
                                  06   12 00283          BNEQ    37$                                           ;
                   53        08   AE   D1 00285          CMPL    SPECIFIED_EXTENT, EXTENT                      ; 0950
                                  1C   13 00289          BEQL    39$                                           ;
        04         0C        AE   01   E1 0028B 37$:     BBC     #1, FLAGS, 38$                                ; 0951
                             13   0C   AE   E9 00290     BLBC    FLAGS, 39$                                    ;
                                  0C   BB 00294 38$:     PUSHR   #^M<R2,R3>                                    ; 0954
                                  54   DD 00296          PUSHL   LIMIT                                         ; 0953
                                  03   DD 00298          PUSHL   #3                                            ; 0952
                       00000000G  8F   DD 0029A          PUSHL   #SET$_NEWLIMS                                 ;
             00000000G  00        05   FB 002A0          CALLS   #5, LIB$SIGNAL                                ;
                                       04 002A7 39$:     RET                                                   ; 0958

; Routine Size:  680 bytes,    Routine Base:  $CODE$ + 04BE
;  969      0959  1 ROUTINE setwrkknl (limit, quota, extent, flags) =
;  970      0960  2 BEGIN
;  971      0961  2 !++
;  972      0962  2 !
;  973      0963  2 ! This is the kernel mode routine that actually sets the working set parameters
;  974      0964  2 !
;  975      0965  2 ! Inputs:
;  976      0966  2 !       LIMIT - address of ws limit
;  977      0967  2 !       QUOTA - address of ws quota
;  978      0968  2 !       EXTENT - address of ws extent
;  979      0969  2 !       FLAGS - address of flags longword
;  980      0970  2 !
;  981      0971  2 ! Outputs:
;  982      0972  2 !       None.  The working set parameters are reset.
;  983      0973  2 !
;  984      0974  2 !--
;  985      0975  2
;  986      0976  2 MAP flags : REF $BBLOCK;
;  987      0977  2
;  988      0978  2 BIND
;  989      0979  2     phd = .ctl$gl_phd : $BBLOCK;                    ! Point to this process's PHD
;  990      0980  2
;  991      0981  2 !
;  992      0982  2 ! Set the values.  Note that all these values are biased by the working set
;  993      0983  2 ! list minus one.  Memory management is the sort of thing that causes one
;  994      0984  2 ! to long for the days of the abacus.
;  995      0985  2 !
;  996      0986  2 phd[phd$w_dfwscnt] = .phd[phd$w_wslist] - 1 + .limit;
;  997      0987  2 phd[phd$w_wsquota] = .phd[phd$w_wslist] - 1 + .quota;
;  998      0988  2 phd[phd$w_wsextent] = .phd[phd$w_wslist] - 1 + .extent;
;  999      0989  2
; 1000      0990  2 !
; 1001      0991  2 ! If the ADJUST qualifier was specified, do it.
; 1002      0992  2 !
; 1003      0993  2 IF .flags[set$v_expadj]
; 1004      0994  2 THEN
; 1005      0995  3     BEGIN
; 1006      0996  3     BIND
; 1007      0997  3         pcb = .ctl$gl_pcb : $BBLOCK;
; 1008      0998  3     pcb[pcb$v_disaws] =  NOT .flags[set$v_adjust];
; 1009      0999  2     END;
; 1010      1000  2
; 1011      1001  2 RETURN 1;
; 1012      1002  1 END;



                                          0000 00000 SETWRKKNL:
                                                               .WORD   Save nothing                  ; 0959
                     50 00000000G  00  D0 00002                MOVL    CTL$GL_PHD, R0                ; 0979
                     51        08  A0  3C 00009                MOVZWL  8(R0), R1                     ; 0986
                     51        04  AC  C0 0000D                ADDL2   LIMIT, R1                     ;
         1A   A0     51            01  A3 00011                SUBW3   #1, R1, 26(R0)                ;
                     51        08  A0  3C 00016                MOVZWL  8(R0), R1                     ; 0987
                     51        08  AC  C0 0001A                ADDL2   QUOTA, R1                     ;
```

```
                 18   A0           51           01  A3 0001E         SUBW3   #1, R1, 24(R0)                           ;
                                   51       08  A0  3C 00023         MOVZWL  8(R0), R1                                ; 0988
                                   51       0C  AC  C0 00027         ADDL2   EXTENT, R1                               ;
                 16   A0           51           01  A3 0002B         SUBW3   #1, R1, 22(R0)                           ;
                      16      10   BC           05  E1 00030         BBC     #5, @FLAGS, 1$                           ; 0993
                                   50 00000000G 00  D0 00035         MOVL    CTL$GL_PCB, R0                           ; 0997
          51     10   BC           01           06  EF 0003C         EXTZV   #6, #1, @FLAGS, R1                       ; 0998
                                   51           51  D2 00042         MCOML   R1, R1                                   ;
     27   A0          01           00           51  F0 00045         INSV    R1, #0, #1, 39(R0)                       ;
                                   50           01  D0 0004B 1$:     MOVL    #1, R0                                   ; 1001
                                                04     0004E         RET                                              ; 1002

; Routine Size:  79 bytes,    Routine Base:  $CODE$ + 0766
; 1014          1003  1 END
; 1015          1004  0 ELUDOM



                                                         .EXTRN  LIB$SIGNAL, LIB$STOP

;                                PSECT SUMMARY
;
;       Name                       Bytes                     Attributes
;
;    $PLIT$                           560  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $CODE$                          1973  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        33         0      1000        00:01.8




;                                COMMAND QUALIFIERS

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

; Size:          1973 code + 560 data bytes
; Run Time:        00:31.9
; Elapsed Time:    01:46.2
; Lines/CPU Min:     1886
; Lexemes/CPU-Min: 18037
; Memory Used:  217 pages
; Compilation Complete
```

0053 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


SETFILE
LIS

SETP0MESS
LIS

SETP0DISP
LIS

SETMISC
LIS

SETPRO
LIS

SETMAIN
LIS