```
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
   SSSSSSSSSSSS  YYY         YYY     SSSSSSSSSSSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS              YYY         YYY  SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
SSS                 YYY   YYY     SSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
   SSSSSSSSS           YYY           SSSSSSSSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
            SSS        YYY                    SSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
SSSSSSSSSSSS           YYY        SSSSSSSSSSSS
```

```
_$

Ps
--
YZ

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$

Z$
```

```
                                                  D 16
**FILE**ID**SYSUPDSEC




 SSSSSSSS   YY      YY   SSSSSSSS   UU      UU  PPPPPPPP    DDDDDDDD      SSSSSSSS   EEEEEEEEEE    CCCCCCCC
 SSSSSSSS   YY      YY   SSSSSSSS   UU      UU  PPPPPPPP    DDDDDDDD      SSSSSSSS   EEEEEEEEEE    CCCCCCCC
SS          YY      YY  SS          UU      UU  PP      PP  DD      DD  SS           EE          CC
SS          YY      YY  SS          UU      UU  PP      PP  DD      DD  SS           EE          CC
SS            YY  YY    SS          UU      UU  PP      PP  DD      DD  SS           EE          CC
SS            YY  YY    SS          UU      UU  PP      PP  DD      DD  SS           EE          CC
 SSSSSS         YY       SSSSSS     UU      UU  PPPPPPPP    DD      DD    SSSSSS     EEEEEEEE    CC
 SSSSSS         YY       SSSSSS     UU      UU  PPPPPPPP    DD      DD    SSSSSS     EEEEEEEE    CC
       SS       YY             SS   UU      UU  PP          DD      DD         SS    EE          CC
       SS       YY             SS   UU      UU  PP          DD      DD         SS    EE          CC
       SS       YY             SS   UU      UU  PP          DD      DD         SS    EE          CC          ....
       SS       YY             SS   UU      UU  PP          DD      DD         SS    EE          CC          ....
SSSSSSSS        YY      SSSSSSSS    UUUUUUUUUU  PP          DDDDDDDD    SSSSSSSS     EEEEEEEEEE    CCCCCCCC  ....
SSSSSSSS        YY      SSSSSSSS    UUUUUUUUUU  PP          DDDDDDDD    SSSSSSSS     EEEEEEEEEE    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
```

Table of contents

```
0000             .TITLE  SYSUPDSEC - Update Section File System Service
0000             .IDENT  'V04-000'
0000        ;
0000        ;*****************************************************************************
0000        ;*                                                                           *
0000        ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000        ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000        ;*  ALL RIGHTS RESERVED.                                                     *
0000        ;*                                                                           *
0000        ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000        ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000        ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000        ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000        ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000        ;*  TRANSFERRED.                                                             *
0000        ;*                                                                           *
0000        ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000        ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000        ;*  CORPORATION.                                                             *
0000        ;*                                                                           *
0000        ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000        ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000        ;*                                                                           *
0000        ;*                                                                           *
0000        ;*****************************************************************************
0000
0000        ;++
0000        ; FACILITY:      UPDATE SECTION SYSTEM SERVICE
0000        ;
0000        ; ABSTRACT:
0000        ;
0000        ; ENVIRONMENT:
0000        ;
0000        ; AUTHOR: PETER H. LIPMAN       , CREATION DATE: 21-APR-78
0000        ;
0000        ; MODIFIED BY:
0000        ;
0000        ;       V03-002 WMC0001         Wayne Cardoza           02-Mar-1983
0000        ;               MMG$CRECOM2 has gone away, MMG$INADRINI returns status
0000        ;
0000        ;       V03-001 SOP0001         J. R. Sopka             27 August 1982
0000        ;               Add XIP_B_MAXACMODE field to IRP extension used by $UPDSEC
0000        ;               and use it for page owner access mode instead of IRP$B_RMOD
0000        ;               which should contan the mode of the requestor.
0000        ;
0000        ;--
```

```
          0000            .SBTTL  DECLARATIONS
          0000    ;
          0000    ; INCLUDE FILES:
          0000    ;
          0000            $ACBDEF                               ;AST control block definitions
          0000            $CADEF                                ;Conditional assembly definitions
          0000            $DYNDEF                               ;Dynamic data structure type codes
          0000            $GSDDEF                               ;Global section descriptor definitions
          0000            $IRPDEF                               ;I/O request packet definitions
          0000            $IPLDEF                               ;Processor priority levels
          0000            $MMGDEF                               ; Offsets from FP into scratch area
          0000            $PCBDEF                               ;Process control block definitions
          0000            $PFNDEF                               ;Page frame number data base definitions
          0000            $PHDDEF                               ;Process header definitions
          0000            $PRDEF                                ;Processor register definitions
          0000            $PRIDEF                               ;Priority increment class definitions
          0000            $PSLDEF                               ;Processor Status Long Word definitions
          0000            $PTEDEF                               ;Page table entry definitions
          0000            $RSNDEF                               ;Resource definitions
          0000            $SECDEF                               ;Section table entry definitions
          0000            $SHBDEF                               ;Shared memory control block definitions
          0000            $SSDEF                                ;System status code definitions
          0000            $VADEF                                ;Virtual address field definitions
          0000    ;
          0000    ; MACROS:
          0000    ;
          0000
          0000    ;
          0000    ; EQUATED SYMBOLS:
          0000    ;
          0000    ; Offset from AP
          0000    ;
00000004  0000            INADR           = 4                   ;Offset to input range
00000008  0000            RETADR          = 8                   ;Offset to return range
0000000C  0000            ACMODE          = 12                  ;Access Mode
00000010  0000            FLAGS           = 16                  ;Flags parameter
00000014  0000            EFN             = 20                  ;QI/O Event Flag
00000018  0000            IOSB            = 24                  ;QI/O I/O Status Block Address
0000001C  0000            ASTADR          = 28                  ;QI/O AST address
00000020  0000            ASTPRM          = 32                  ;QI/O AST parameter
          0000    ;
          0000    ; Offsets into I/O packet while being used as scratch storage for clustering
          0000    ;
          0000            $OFFSET 0,POSITIVE,<-
          0000                    SVAPTE,-                      ;Master page table entry address
          0000                    PTEDAT,-                      ;Process PTE data
          0000                    <,3>,-
          0000                    <IRP_RMOD,1>,-                ;Request mode
          0000                    MFYCNT,-                      ;Cluster count at last modified page
          0000                    IRP_AST,-                     ;Ast address
          0000                    IRP_ASTPRM,-                  ;Ast parameter
          0000                    CLUSTER,-                     ;Maximum size of cluster to scan for
          0000                    COUNT,-                       ;Number of pages scanned
          0000                    <EXCLWRT,1>,-                 ;Exclusive write access flag
          0000                    <,1>,-
          0000                    <IRP_EFN,1>,-                 ;Event flag
          0000                    <IRP_PRI,1>,-                 ;Priority
```

```
0000                                 IRP_IOSB,-                  ;I/O status block address
0000                                 INCT,-                      ;+ or - 1 according to direction
0000                                 INC4,-                      ;+ or - 4 according to direction
0000                                 BAK,-                       ;Backing store address of first PTE
0000                                 <,4>,-
0000                                 <IRP_IOST1,8>,-             ;I/O status return area
0000                                 PROCPTE,-                   ;Process page table entry address
0000                                 <,4>,-
0000                                 IRP_SEGVBN,-                ;Starting virtual address of scan
0000                                 <IRP_LENGTH,0>-             ;Total size of scratch area used
0000                                 >
0000           SVAPTE:
0004           PTEDAT:
000B           IRP_RMOD:
000C           MFYCNT:
0010           IRP_AST:
0014           IRP_ASTPRM:
0018           CLUSTER:
001C           COUNT:
0020           EXCLWRT:
0022           IRP_EFN:
0023           IRP_PRI:
0024           IRP_IOSB:
0028           INCT:
002C           INC4:
0030           BAK:
0038           IRP_IOST1:
0040           PROCPTE:
0048           IRP_SEGVBN:
004C           IRP_LENGTH:
0000
0000                   ASSUME  IRP_LENGTH      LE IRP$C_LENGTH
0000                   ASSUME  IRP_RMOD        EQ IRP$B_RMOD
0000                   ASSUME  IRP_AST         EQ IRP$L_AST
0000                   ASSUME  IRP_ASTPRM      EQ IRP$L_ASTPRM
0000                   ASSUME  IRP_EFN         EQ IRP$B_EFN
0000                   ASSUME  IRP_PRI         EQ IRP$B_PRI
0000                   ASSUME  IRP_IOSB        EQ IRP$L_IOSB
0000                   ASSUME  IRP_IOST1       EQ IRP$L_IOST1
0000                   ASSUME  IRP_SEGVBN      EQ IRP$L_SEGVBN
0000         ;
0000         ; Offsets off the end of the I/O request packet
0000         ;
0000                   $OFFSET IRP$C_LENGTH,POSITIVE,<-
0000                                 XIP_L_SCANCNT,-             ;Count - 1 of pages remaining to scan
0000                                 XIP_L_DIREC,-               ;+ OR - 200 according to the direction
0000                                 XIP_L_STARTVA,-             ;Starting virtual address to scan
0000                                 <XIP_B_UPDFLG,1>,-          ;Section update flags
0000                                 <XIP_B_MAXACMODE,1>,-       ;Maximzed access mode for page ownership
0000                                 <,2>,-                      ;Spare
0000                                 <XIP_C_LENGTH,0> -          ;Length of extended I/O packet
0000                                 >
00C4           XIP_L_SCANCNT:
00C8           XIP_L_DIREC:
00CC           XIP_L_STARTVA:
00D0           XIP_B_UPDFLG:
00D1           XIP_B_MAXACMODE:
```

```
                    00D4        XIP_C_LENGTH:
                    0000   138  ;
                    0000   139  ; OWN STORAGE:
                    0000   140  ;
                    0000   141          .LIST   MEB
```

```
0000          .SBTTL  UPDSEC - Update Section File
0000    ;++
0000    ; FUNCTIONAL DESCRIPTION:
0000    ;
0000    ; CALLING SEQUENCE:
0000    ;
0000    ;       CALLG   ARGLIST,G^SYS$UPDSEC
0000    ;
0000    ;
0000    ; INPUT PARAMETERS:
0000    ;
0000    ;       INADR(AP) = Address of 2 long words the 1st of which specifies
0000    ;               the starting virtual address, the 2nd specifies the ending
0000    ;               virtual address (inclusive) of the pages to operate on.
0000    ;       RETADR(AP) = Address of a 2 longword array into which is returned
0000    ;               the starting and ending virtual addresses (inclusive)
0000    ;               of the pages operated on.
0000    ;       ACMODE(AP) = The access mode (maximized with calling mode)
0000    ;               against which the page ownership is checked.
0000    ;               Only the owner of a page may update its section.
0000    ;       FLAGS(AP)  = Update section control flags
0000    ;       EFN(AP)    = Event flag number to set on write complete
0000    ;       IOSB(AP)   = I/O status block address for reporting the
0000    ;                    write completion and its status
0000    ;                       First word contains the system status.
0000    ;                       If error status is returned in the first word,
0000    ;                    the first bit of the 2nd word (bit 16 of the first
0000    ;                    long word) will be set if a write error occurred.
0000    ;                    Other errors (e.g. page owner violation) are possible.
0000    ;                       The second long word contains the first virtual
0000    ;                    address not written.
0000    ;       ASTADR(AP) = AST address for reporting write completion
0000    ;       ASTPRM(AP) = AST parameter for identifying the AST
0000    ;
0000    ; IMPLICIT INPUTS:
0000    ;
0000    ;       NONE
0000    ;
0000    ; OUTPUT PARAMETERS:
0000    ;
0000    ;       R0 = System Status Code
0000    ;
0000    ; IMPLICIT OUTPUTS:
0000    ;
0000    ;       NONE
0000    ;
0000    ; COMPLETION CODES:
0000    ;
0000    ;       SS$_NORMAL                              ;Successful Completion
0000    ;       SS$_ACCVIO                              ;Access Violation
0000    ;       SS$_PAGOWNVIO                           ;Page Owner Violation
0000    ;       SS$_EXQUOTA                             ;Quota exceeded for pending AST's
0000    ;       SS$_IVSECFLG                            ;Invalid flags set
0000    ;
0000    ; SIDE EFFECTS:
0000    ;
0000    ;       NONE
```

```
                              0000   ;
                              0000   ;--
                              0000   ;
                              0000   ; ***********************************************************************
                              0000   ;
                              0000   ; **************** THE FOLLOWING CODE MAY BE PAGED *****************
                              0000   ;
                          00000000          .PSECT  Y$EXEPAGED
                              0000   ;
                              0000   ; ***********************************************************************
                              0000   ;
                              0000
                              0000   INADRERR:
                        04    0000          RET
                              0001
                      01FC    0001          .ENTRY  EXE$UPDSEC,^M<R2,R3,R4,R5,R6,R7,R8>
                              0003
          5E    1C    C2      0003          SUBL    S^#-MMG$C_LENGTH,SP        ;Reserve area indexed from FP
          58    54    D0      0006          MOVL    R4,R8                      ;Save PCB address
              FFF4'   30      0009          BSBW    MMG$INADRINI               ;Get input address range to R4,R5
                              000C                                             ;Init return range to null
             F1 50    E9      000C          BLBC    R0,INADRERR
                30    BB      000F          PUSHR   #^M<R4,R5>                 ;Save input address range
          54    58    D0      0011          MOVL    R8,R4                      ;Restore PCB address
       53    14 AC    9A      0014          MOVZBL  EFN(AP),R3                 ;Get the event flag parameter
      00000000'EF     16      0018          JSB     SCH$CLREF                  ;Clear the specified event flag
       56    18 AC    D0      001E          MOVL    IOSB(AP),R6                ;Get I/O status block address
                08    13      0022          BEQL    20$                        ;Branch if none specified
                              0024          IFNOWRT #8,(R6),70$                ;Make sure caller could write it
    66    08    00    0D      0024                  PROBEW  #0,#8,(R6)
                7F    13      0028                  BEQL    70$
                66    7C      002A          CLRQ    (R6)                       ;and initialize it
       57    10 AC    D0      002C   20$:   MOVL    FLAGS(AP),R7               ;Get FLAGS parameter
          01    57    D1      0030          CMPL    R7,#1                      ;Make sure no garbage bits are set
                6D    1A      0033          BGTRU   60$                        ;Branch if invalid section flags
   51    000000D4 8F  D0      0035          MOVL    #XIP_C_LENGTH,R1           ;Size of packet to allocate
      00000000'EF     16      003C          JSB     EXE$ALLOCBUF               ;Allocate, wait if necessary
                              0042                                             ;Packet type is corrected by WRTPGSBAK
             67 50    E9      0042          BLBC    R0,80$                     ;Branch if failed to alloc
                              0045                                             ;and resource wait disabled
                              0045   ;
                              0045   ; IPL = ASTDEL, I/O request packet allocated
                              0045   ;
          58    52    D0      0045          MOVL    R2,R8                      ;Packet address to stable registter
       52    3E A4    9E      0048          MOVAB   PCB$W_DIOCNT(R4),R2        ;Check for Direct I/O quota
      00000000'EF     16      004C          JSB     EXE$SNGLEQUOTA             ;and wait if none available
             5B 50    E9      0052          BLBC    R0,120$                    ;Branch if exceeded quota
                              0055                                             ;and resource wait is disabled
                50    DC      0055          MOVPSL  R0                         ;Get mode of the requestor
50    50    02    16    EF    0057          EXTZV   #PSL$V_PRVMOD,#PSL$S_PRVMOD,R0,R0
             FC AD    90      005C          MOVB    B^MMG$[_MAXACMODE(FP),-  ;Get maximized access mode
           00D1 C8            005F                  XIP_B_MAXACMODE(R8)        ; for page ownership checking
    10 A8    1C AC    7D      0062          MOVQ    ASTADR(AP),IRP$L_AST(R8)   ;Set AST address and parameter
             10 A8    D5      0067          TSTL    IRP$L_AST(R8)              ;AST requested?
                0C    13      006A          BEQL    40$                        ;Branch if not
             38 A4    B5      006C          TSTW    PCB$W_ASTCNT(R4)           ;Yes, quota exceeded?
                3F    15      006F          BLEQ    120$                       ;Branch if yes, don't wait
          38 A4   B7  0071              DECW     PCB$W_ASTCNT(R4)              ;Charge for the AST
     50   40 8F   88  0074              BISB     #ACB$M_QUOTA,R0               ;And note that it is charged
       0B A8  50  90  0078        40$:  MOVB     R0,IRP$B_RMOD(R8)             ;Set requesting mode and AST flag
    22 A8  14 AC  90  007C              MOVB     EFN(AP),IRP$B_EFN(R8)         ;Set event flag number
       24 A8  56  D0  0081              MOVL     R6,IRP$L_IOSB(R8)             ;Set I/O status block address
     00D0 C8  57  90  0085              MOVB     R7,XIP_B_UPDFLG(R8)           ;Set section update flags
       56   E9'AF 9E  008A              MOVAB    B^MMG$UPDSECPAG,R6            ;Address of per page subroutine
             0C   BA  008E              POPR     #^M<R2,R3>                    ;Recover saved input address range
          FF6D'   30  0090              BSBW     MMG$CREDEL                    ;Common address range loop
             50   DD  0093              PUSHL    R0                            ;Save status
          FF68'   30  0095              BSBW     MMG$RETRANGE
          02 50   E9  0098              BLBC     R0,45$                        ;Use this bad status rather than CREDEL
             50   BA  009B              POPR     R0
             58   D5  009D        45$:  TSTL     R8                            ;I/O packet to be released?
             12   12  009F              BNEQ     130$                          ;Branch if yes
             04       00A1        50$:  RET                                    ;Write was queued successfully
                      00A2
  50   016C 8F    3C  00A2        60$:  MOVZWL   #SS$_IVSECFLG,R0              ;Invalid section flags parameter
             03   11  00A7              BRB      80$
     50      0C   3C  00A9        70$:  MOVZWL   #SS$_ACCVIO,R0                ;Access violation
             50   DD  00AC        80$:  PUSHL    R0                            ;Save the status code
             16   11  00AE              BRB      140$
                      00B0        ;
                      00B0        ; Release the I/O request packet, it was never used
                      00B0        ;
     50      1C   3C  00B0       120$:  MOVZWL   #SS$_EXQUOTA,R0               ;Exceeded quota
             50   DD  00B3       130$:  PUSHL    R0                            ;Save status
  03 0B A8   06   E5  00B5              BBCC     #ACB$V_QUOTA,IRP$B_RMOD(R8),135$ ;If charged for AST
          38 A4   B6  00BA              INCW     PCB$W_ASTCNT(R4)              ;then give back the quota
       50    58   D0  00BD       135$:  MOVL     R8,R0                         ;Get I/O packet address to release
   00000000'EF    16  00C0              JSB      EXE$DEANONPAGED               ;Release the I/O request packet
                      00C6        ;
                      00C6        ; Set the event flag so that the caller may wait for it despite the return
                      00C6        ; information showing that nothing was queued.
                      00C6        ;
    53     14 AC  9A  00C6       140$:  MOVZBL   EFN(AP),R3                    ;Get the event flag number
    51     60 A4  D0  00CA              MOVL     PCB$L_PID(R4),R1              ;and the process ID
       52    01   9A  00CE              MOVZBL   #PRI$_IOCOM,R2                ;and the correct priority increment
   00000000'EF    16  00D1              JSB      SCH$POSTEF                    ;Post the event flag, write complete
             01   BA  00D7              POPR     #^M<R0>                       ;Restore saved status
    51     18 AC  D0  00D9              MOVL     IOSB(AP),R1                   ;I/O status requested?
             09   13  00DD              BEQL     150$                          ;Branch if not
                      00DF              IFNOWRT  #8,(R1),150$                  ;Branch if IOSB not writable
   61    08   00  0D  00DF                       PROBEW  #0,#8,(R1)
             03   13  00E3                       BEQL    150$
      61     50   D0  00E5              MOVL     R0,(R1)                       ;Return the error status
             04       00E8       150$:  RET                                    ;and return
    00E9         .SBTTL  UPDSECPAG - Update Section for First Cluster of Pages
    00E9   ;
    00E9   ; ***********************************************************************
    00E9   ;
    00E9   ; **************** THE FOLLOWING CODE MAY BE PAGED ****************
    00E9   ;
000000E9         .PSECT  Y$EXEPAGED
    00E9   ;
    00E9   ; ***********************************************************************
    00E9   ;
    00E9   ;++
    00E9   ; FUNCTIONAL DESCRIPTION:
    00E9   ;
    00E9   ;
    00E9   ; CALLING SEQUENCE:
    00E9   ;
    00E9   ;        BSBW    MMG$UPDSECPAG
    00E9   ;
    00E9   ;
    00E9   ; INPUT PARAMETERS:
    00E9   ;
    00E9   ;        R0 = Access Mode for page ownership check
    00E9   ;        R2 = Virtual Address
    00E9   ;        R4 = Current PCB address
    00E9   ;        R5 = Process Header Address - P1 or System Space
    00E9   ;        R6 = Count - 1 of pages to be processed including this one
    00E9   ;        R7 = +^X200 if going forward in the address space
    00E9   ;           = -^X200 if going backwards in the address space
    00E9   ;        R8 = Address of an extended length I/O request packet
    00E9   ;                IRP$W_SIZE      = size of extended IRP (XIP_C_LENGTH)
    00E9   ;                                  type filled in by WRTPGSBAK
    00E9   ;                IRP$L_ASTADR    = AST address if desired
    00E9   ;                IRP$L_ASTPRM    = AST parameter
    00E9   ;                IRP$B_RMOD      = Requesting mode
    00E9   ;                                  ACB$V_QUOTA set if AST desired
    00E9   ;                IRP$B_EFN       = Event flag number
    00E9   ;                XIP_L_DIREC     = + OR - ^X200 according to direction of scan
    00E9   ;                XIP_B_UPDFLG    = Update section flags
    00E9   ;
    00E9   ;        IPL = ASTDEL
    00E9   ;
    00E9   ; IMPLICIT INPUTS:
    00E9   ;        NONE
    00E9   ;
    00E9   ; OUTPUT PARAMETERS:
    00E9   ;
    00E9   ;        R0 = Status Code
    00E9   ;        R2   Preserved
    00E9   ;
    00E9   ; IMPLICIT OUTPUTS:
    00E9   ;        NONE
    00E9   ;
    00E9   ; COMPLETION CODES:
    00E9   ;
    00E9   ;        SS$_NORMAL                          ;Successful Completion
    00E9   ;        SS$_PAGOWNVIO                       ;Page Owner Violation
    00E9   ;        SS$_LENVIO                          ;Length Violation
```

```
                          00E9     ;           SS$_ACCVIO                     ;Access Violation
                          00E9     ;
                          00E9     ; SIDE EFFECTS:
                          00E9     ;
                          00E9     ;           NONE
                          00E9     ;
                          00E9     ;--
                          00E9
                          00E9     MMG$UPDSECPAG:
       00C8 C8   57   D0  00E9             MOVL    R7,XIP_L_DIREC(R8)          ;Save direction of scan
        00000000'EF   16  00EE             JSB     MMG$UPDSECQWT               ;Find and queue the next cluster
                 51   D5  00F4             TSTL    R1                          ;Anything queued for writing?
                 0D   12  00F6             BNEQ    20$                         ;Branch if yes
              F4 AD   D4  00F8             CLRL    B^MMG$L_SAVRETADR(FP)       ;Return a null range
              15 50   E9  00FB             BLBC    R0,60$                      ;Branch if error status
    50     0659 8F    3C  00FE             MOVZWL  #SS$_NOTMODIFIED,R0         ;Otherwise return alternate success code
                 0E   11  0103             BRB     60$
                 58   D4  0105     20$:    CLRL    R8                          ;Note I/O packet in use
         EC AD   52   D0  0107             MOVL    R2,B^MMG$L_SVSTARTVA(FP)    ;Return first address queued
                 51   D7  010B             DECL    R1                          ;Page count - 1
            51   57   C4  010D             MULL    R7,R1                       ;Byte count
            52   51   C0  0110             ADDL    R1,R2                       ;Address of last page queued
                 56   D4  0113     60$:    CLRL    R6                          ;Force end of range
                 05       0115             RSB                                 ;and return
```

```
0116            .SBTTL  UPDSECAST - Update Section AST
0116    ;++
0116    ; FUNCTIONAL DESCRIPTION:
0116    ;
0116    ;       This is a special kernel AST routine invoked by IOPOST at the
0116    ; completion of a PAGIO write request with an extended I/O packet.
0116    ; It's job is to find the next cluster of modified pages to write
0116    ; and either queue the request or post the I/O completion.
0116    ;
0116    ; CALLING SEQUENCE:
0116    ;
0116    ;       BSBW    MMG$UPDSECAST
0116    ;
0116    ;
0116    ; INPUT PARAMETERS:
0116    ;
0116    ;       R4 = Current PCB address
0116    ;       R5 = Address of an extended length I/O request packet
0116    ;               IRP$W_SIZE      = size of extended IRP (XIP_C_LENGTH)
0116    ;               IRP$B_TYPE      = DYN$C_IRP
0116    ;               IRP$L_ASTADR    = AST address if desired
0116    ;               IRP$L_ASTPRM    = AST parameter
0116    ;               IRP$B_RMOD      = Requesting mode
0116    ;                                 ACB$V_QUOTA set if AST desired
0116    ;               IRP$B_EFN       = Event flag number
0116    ;               XIP_L_SCANCNT   = Count - 1 of pages left to scan
0116    ;                                 before this transfer completed
0116    ;               XIP_L_DIREC     = + OR - ^X200 according to direction of scan
0116    ;               XIP_L_STARTVA   = First VA used for this transfer
0116    ;               XIP_B_UPDFLG    = Update section flags
0116    ;               XIP_B_MAXACMODE = Maximized access mode for page ownership
0116    ;               IPR$L_IOST1     = Status of previous write (0:15)
0116    ;                               = Number of bytes successfully written (16:31)
0116    ;
0116    ;       IPL = ASTDEL
0116    ;
0116    ; IMPLICIT INPUTS:
0116    ;       NONE
0116    ;
0116    ; OUTPUT PARAMETERS:
0116    ;
0116    ;
0116    ; IMPLICIT OUTPUTS:
0116    ;       NONE
0116    ;
0116    ; COMPLETION CODES:
0116    ;
0116    ;
0116    ; SIDE EFFECTS:
0116    ;
0116    ;       NONE
0116    ;
0116    ;--
```

```
                          0116  ;
                          0116  ; ************************************************************************
                          0116  ;
                          0116  ; **************** THE FOLLOWING CODE MAY BE PAGED *****************
                          0116  ;
                      00000116          .PSECT  Y$EXEPAGED
                          0116  ;
                          0116  ; ************************************************************************
                          0116  ;
                          0116
                          0116  MMG$UPDSECAST::
             01C0 8F   BB 0116          PUSHR   #^M<R6,R7,R8>           ;Save these registers
                58 55  D0 011A          MOVL    R5,R8                   ;I/O request packet address
    55   00000000'GF   D0 011D          MOVL    G^CTL$GL_PHD,R5         ;Get P1 address of process header
                          0124
                          0124          ASSUME  XIP_L_DIREC EQ XIP_L_SCANCNT+4
        56  00C4 C8    7D 0124          MOVQ    XIP_L_SCANCNT(R8),R6    ;R6=count-1, R7=+ or - ^X200
        52  00CC C8    D0 0129          MOVL    XIP_L_STARTVA(R8),R2    ;R2 = first VA of this transfer
           50  38 A8   D0 012E          MOVL    IRP$L_IOST1(R8),R0      ;Get status and byte count
    51     50  07 19   EF 0132          EXTZV   #<16+VA$V_VPN>,#<16-VA$V_VPN>,R0,R1 ;Page count transferred
        53     51 57   C5 0137          MULL3   R7,R1,R3                ;Directional byte count
               52 53   C0 013B          ADDL    R3,R2                   ;New starting VA = first VA not written
            3C A8 52   D0 013E          MOVL    R2,IRP$L_IOST2(R8)      ;Save it as second IOSB long word
               17 50   E9 0142          BLBC    R0,100$                 ;Branch if write error
           56     51   C2 0145          SUBL    R1,R6                   ;Page count remaining to scan
                  12   19 0148          BLSS    100$                    ;Branch if did last piece
         00000000'EF   16 014A          JSB     MMG$UPDSECQWT           ;Scan for another cluster to write
               09 50   E9 0150          BLBC    R0,100$                 ;Dont continue scanning if error, branch
                  51   D5 0153          TSTL    R1                      ;Anything found and queued?
                  05   13 0155          BEQL    100$                    ;Branch if not
             01C0 8F   BA 0157          POPR    #^M<R6,R7,R8>           ;restore saved registers
                  05      015B          RSB                             ;and return from AST
                          015C  ;
                          015C  ; Last cluster of pages was written
                          015C  ; R0 = status
                          015C  ;
               55 58   D0 015C  100$:   MOVL    R8,R5                   ;I/O packet address back to R5
             01C0 8F   BA 015F          POPR    #^M<R6,R7,R8>           ;Restore registers
               50 50   3C 0163          MOVZWL  R0,R0                   ;Zero high 16 bits of status
            04 38 A5   E8 0166          BLBS    IRP$L_IOST1(R5),120$    ;Branch if not page write error
            00 50 10   E2 016A          BBSS    #16,R0,120$             ;Set page write error indication
            38 A5 50   D0 016E  120$:   MOVL    R0,IRP$L_IOST1(R5)      ;Set first long word of return status
            53 22 A5   9A 0172          MOVZBL  IRP$B_EFN(R5),R3        ;Get the event flag to post
            51 0C A5   D0 0176          MOVL    IRP$L_PID(R5),R1        ;Process ID
               52 01   9A 017A          MOVZBL  #PRI$_IOCOM,R2          ;Priority increment for I/O completion
         00000000'EF   16 017D          JSB     SCH$POSTEF              ;Post the event flag
         00000000'EF   17 0183          JMP     IOC$DIRPOST1            ;Go return status to IOSB if specified
                          0189                                          ;and issue AST if requested
```

```
SYS
Sym

ACB
ACB
ACM
AST
AST
BAK
BUG
BUG
CA$
CLU
COU
CTL
DIR
DYN
EFN
EXC
EXE
EXE
EXE
EXE
EXE
FLA
GSD
GSD
GSD
GSD
GSD
INA
INA
INC
INC
IOC
IOS
IPL
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
IRP
MFY
```

```
0189          .SBTTL  UPDSECQWT - Update Section File for Single Page
0189
0189    ;++
0189    ; FUNCTIONAL DESCRIPTION:
0189    ;
0189    ;
0189    ; CALLING SEQUENCE:
0189    ;
0189    ;       BSBW    MMG$UPDSECQWT
0189    ;
0189    ;
0189    ; INPUT PARAMETERS:
0189    ;
0189    ;       R2 = Virtual Address
0189    ;       R4 = Current PCB address
0189    ;       R5 = Process Header Address - P1 or System Space
0189    ;       R6 = Count - 1 of pages to be processed including this one
0189    ;       R7 = +^X200 if going forward in the address space
0189    ;          = -^X200 if going backwards in the address space
0189    ;       R8 = Address of an extended length I/O request packet
0189    ;               IRP$W_SIZE        = size of extended IRP (XIP_C_LENGTH)
0189    ;                                   type filled in by WRTPGSBAK
0189    ;               IRP$L_ASTADR      = AST address if desired
0189    ;               IRP$L_ASTPRM      = AST parameter
0189    ;               IRP$B_RMOD        = Requesting mode
0189    ;                                   ACB$V_QUOTA set if AST desired
0189    ;               IRP$B_EFN         = Event flag number
0189    ;               XIP_L_DIREC       = + OR - ^X200 according to direction of scan
0189    ;               XIP_B_UPDFLG      = Update section flags
0189    ;               XIP_B_MAXACMODE = Maximized access mode for page ownership
0189    ;
0189    ;       IPL = ASTDEL
0189    ;
0189    ; IMPLICIT INPUTS:
0189    ;       NONE
0189    ;
0189    ; OUTPUT PARAMETERS:
0189    ;
0189    ;    If write has been queued, then
0189    ;
0189    ;       R0 = #SS$_NORMAL
0189    ;       R1 = number of pages queued for writing
0189    ;       R2 = virtual address of first page (scan order) queued
0189    ;       R6 = count - 1 of pages remaining to scan starting with VA in R2;
0189    ;
0189    ;       Extended portion of I/O request packet updated if write queued
0189    ;               XIP_L_STARTVA   = starting virtual address of request just queued
0189    ;               XIP_L_SCANCNT   = count - 1 of pages remaining to scan
0189    ;                                 starting with the first page just queued
0189    ;
0189    ;    If write has not been queued, then
0189    ;
0189    ;       R0 = system status code
0189    ;       R1 = 0
0189    ;       R2 = last virtual address scanned
0189    ;            in the case of an error, this is the address that caused it
0189    ;            if ran off the end of range, this is the last VA in the range
```

```
    0189  ;       R6 = count - 1 of pages remaining to scan starting with VA in R2
    0189  ;          = 0 if at end of range and no more to do
    0189  ;
    0189  ; IMPLICIT OUTPUTS:
    0189  ;
    0189  ;       NONE
    0189  ;
    0189  ; COMPLETION CODES:
    0189  ;
    0189  ;       SS$_NORMAL                              ;Successful Completion
    0189  ;       SS$_PAGOWNVIO                           ;Page Owner Violation
    0189  ;       SS$_LENVIO                              ;Length Violation
    0189  ;       SS$_ACCVIO                              ;Access Violation
    0189  ;
    0189  ; SIDE EFFECTS:
    0189  ;
    0189  ;       NONE
    0189  ;
    0189  ;--
    0189  ;
    0189  ; *************************************************************************
    0189  ;
    0189  ; **************** THE FOLLOWING CODE MUST BE RESIDENT ****************
    0189  ;
00000000          .PSECT  $MMGCOD
    0000  ;
    0000  ; *************************************************************************
    0000  ;
```

```
SYS
VAX

Pha
---
Ini
Com
Pas
Sym
Pas
Sym
Pse
Cro
Ass

The
947
The
105
36

Mac
---
_$2
_$2
TOT

159

The

MAC
```

```
                             0000  MMG$UPDSECQWT:
              51     D4      0000             CLRL     R1                        ;Initialize indicator to no pages queued
            FFFB'    30      0002             BSBW     MMG$PTEINDX               ;Get index to page table entry
           64 50     E9      0005             BLBC     R0,100$                   ;Branch if length violation
                             0008             DSBINT   #IPL$_SYNCH               ;Push current IPL
          7E  12     DB      0008                      MFPR    S^#PR$_IPL,-(SP)
          12  08     DA      000B                      MTPR    #IPL$_SYNCH,S^#PR$_IPL
                             000E                                                ;and raise to SYNCH
       53  6C B443   DE      000E             MOVAL    @PCB$L_PHD(R4)[R3],R3     ;Form system virtual address of PTE
  51    57   F9 8F   78      0013             ASHL     #-7,R7,R1                 ;+ OR - 4 for adding to  SVAPTE
              0E     BB      0018  10$:       PUSHR    #^M<R1,R2,R3>
              52     D4      001A             CLRL     R2                        ;PTEPFNMFY should return section/GPTX
    50   00D1 C8     9A      001C             MOVZBL   XIP_B_MAXACMODE(R8),R0    ;Access mode to check against page owner
    51   00D0 C8     9A      0021             MOVZBL   XIP_B_UPDFLG(R8),R1       ;Exclusive writer indication
            0254     30      0026             BSBW     MMG$PTEPFNMFY             ;Get PFN and modify bit for this PTE
           06 51     E9      0029             BLBC     R1,20$                    ;Branch if page not a candidate for write
              51     95      002C             TSTB     R1                        ;Could be written, is it modified?
              21     19      002E             BLSS     70$                       ;Branch if yes, go write a cluster
              05     11      0030             BRB      30$                       ;No, try the next page if any
          50  51     D0      0032  20$:       MOVL     R1,R0                     ;Error, or just not a candidate?
              14     12      0035             BNEQ     60$                       ;Branch if error
              0E     BA      0037  30$:       POPR     #^M<R1,R2,R3>             ;R3=SVAPTE, R2=VA, R1=+ or - 4
              56     D5      0039             TSTL     R6                        ;Check for end of loop
              09     13      003B             BEQL     40$                       ;Avoid modifying VA and Count
          52  57     C0      003D             ADDL     R7,R2                     ;Next virtual address
          53  51     C0      0040             ADDL     R1,R3                     ;and next PTE address
           D2 56     F4      0043             SOBGEQ   R6,10$                    ;Try the next page
          50  01     3C      0046  40$:       MOVZWL   #SS$_NORMAL,R0            ;End of range, no more to do
              02     11      0049             BRB      65$
              0E     BA      004B  60$:       POPR     #^M<R1,R2,R3>
              51     D4      004D  65$:       CLRL     R1                        ;No pages queued for writing
              18     11      004F             BRB      80$
                             0051  ;
                             0051  ; Found a page to start the cluster, queue a cluster of pages
                             0051  ;
              02     BA      0051  70$:       POPR     #^M<R1>                   ;Clean off + or - 4
    00CC C8   6E     D0      0053             MOVL     (SP),XIP_L_STARTVA(R8)    ;Save starting VA for UPDSECAST
       48 A8  6E     D0      0058             MOVL     (SP),IRP$L_SEGVBN(R8)     ;and for WRTPGSBAK
    00C4 C8   56     D0      005C             MOVL     R6,XIP_L_SCANCNT(R8)      ;and remaining count for this write
          51  58     D0      0061             MOVL     R8,R1                     ;I/O request packet (extended)
            0006     30      0064             BSBW     MMG$WRTPGSBAK             ;Queue a cluster for write back
              0C     BA      0067             POPR     #^M<R2,R3>                ;Restore saved VA, clean off SVAPTE
                             0069  80$:       ENBINT                             ;Back to called IPL
          12  8E     DA      0069                      MTPR    (SP)+,S^#PR$_IPL
                     05      006C  100$:      RSB
```

```
006D            .SBTTL  WRTPGSBAK - Write Pages Back to Disk
006D    ;++
006D    ; FUNCTIONAL DESCRIPTION:
006D    ;
006D    ;
006D    ; CALLING SEQUENCE:
006D    ;
006D    ;       BSBW    MMG$WRTPGSBAK
006D    ;
006D    ;
006D    ; INPUT PARAMETERS:
006D    ;
006D    ;       R0 = Page Frame Number of starting page
006D    ;       R1 = Address of an I/O request packet
006D    ;               IRP$W_SIZE      = XIP_C_LENGTH if called by UPDSEC
006D    ;                               = IRP$C_LENGTH if called by DELPAG
006D    ;               IRP$B_TYPE      = type filled in by WRTPGSBAK
006D    ;               IRP$L_ASTADR    = AST address if desired
006D    ;               IRP$L_ASTPRM    = AST parameter
006D    ;               IRP$B_RMOD      = Requesting mode
006D    ;                                 ACB$V_QUOTA set if AST desired
006D    ;               IRP$B_EFN       = Event flag number
006D    ;               IRP$L_SEGVBN    = Starting virtual address of scan
006D    ;               XIP_B_UPDFLG    = Update section flags (if extended packet)
006D    ;               XIP_B_MAXACMODE = Maximized access mode for page ownership
006D    ;       R2 = Section backing store address (PFN$AL_BAK[R0])
006D    ;               if process section page or shared memory global page
006D    ;          = Global page table index if global page
006D    ;       R3 = System virtual address of process page table entry for first page
006D    ;       R4 = PCB address
006D    ;       R5 = Process header address - P1 or System Space
006D    ;       R6 = Count - 1 of pages remaining to be processed including this one
006D    ;       R7 = +^X200 if going forward in address space
006D    ;          = -^X200 if going backwards in address space
006D    ;       IPL = SYNCH
006D    ;
006D    ; IMPLICIT INPUTS:
006D    ;
006D    ;       NONE
006D    ;
006D    ; OUTPUT PARAMETERS:
006D    ;
006D    ;       R0 = #SS$_NORMAL
006D    ;       R1 = Number of pages queued for writing
006D    ;       R2,R3 Scratched
006D    ;
006D    ; IMPLICIT OUTPUTS:
006D    ;       NONE
006D    ;
006D    ; COMPLETION CODES:
006D    ;
006D    ;
006D    ; SIDE EFFECTS:
006D    ;
006D    ;
006D    ;--
```

```
                              006D   ;
                              006D   ; ****************************************************************************
                              006D   ;
                              006D   ; **************** THE FOLLOWING CODE MUST BE RESIDENT *****************
                              006D   ;
                          0000006D           .PSECT  $MMGCOD
                              006D   ;
                              006D   ; ****************************************************************************
                              006D   ;
                              006D   MMG$WRTPGSBAK::
                    30    BB  006D           PUSHR   #^M<R4,R5>                 ;Preserve R4 and R5 across call
                              006F   ;
                              006F   ; Initialize I/O packet for cluster scan
                              006F   ;
      2C A1    57    F9 8F    78  006F           ASHL    #-7,R7,INC4(R1)            ;+ or - 4 according to direction
      28 A1    57    F7 8F    78  0075           ASHL    #-9,R7,INC1(R1)            ;+ or - 1 according to direction
                 57    51     D0  007B           MOVL    R1,R7                      ;Packet address in stable register
            0A A7      0A     90  007E           MOVB    #DYN$C_IRP,IRP$B_TYPE(R7)  ;Set packet type, size already set
     00000000'EF       50     D1  0082           CMPL    R0,MMG$GL_MAXPFN           ;Is page in shared memory?
                       32     1A  0089           BGTRU   50$                        ;Br if page is in shared memory gbl sec.
   30 A7     0000'DF40        D0  008B           MOVL    @W^PFN$AL_BAK[R0],BAK(R7)  ;Actual section backing store
                                  0092                                              ;address even if global page
      67     0000'DF40        D0  0092           MOVL    @W^PFN$AL_PTE[R0],SVAPTE(R7) ;Master PTE address even if global
            40 A7      53     D0  0098   30$:    MOVL    R3,PROCPTE(R7)             ;Keep process pte address
            04 A7      52     D0  009C           MOVL    R2,PTEDAT(R7)              ;Save section adr/GPTX
      23 A7      2F A4        90  00A0           MOVB    PCB$B_PRIB(R4),IRP$B_PRI(R7) ;Set transfer priority
                                  00A5   ;
                                  00A5   ; Calculate largest cluster size as the minimum of the default cluster
                                  00A5   ; size and the number of pages left to operate on.
                                  00A5   ;
         51      0000'CF      3C  00A5           MOVZWL  W^MPW$GW_MPWPFC,R1         ;Default cluster size
                 51    56     D1  00AA           CMPL    R6,R1                      ;If count-1 is smaller
                       04     18  00AD           BGEQ    40$
          51     01    A6     DE  00AF           MOVAL   1(R6),R1                   ;then use count as max cluster size
            18 A7      51     D0  00B3   40$:    MOVL    R1,CLUSTER(R7)             ;Set maximum cluster size
            1C A7      01     D0  00B7           MOVL    #1,COUNT(R7)               ;Count the first page in the cluster
                       3C     11  00BB           BRB     80$                        ;and loop zero or more times
                                  00BD   ;
                                  00BD   ; Shared Memory global section pages have no PFN data base.
                                  00BD   ;
            30 A7      52     D0  00BD   50$:    MOVL    R2,BAK(R7)                 ;Use section table index
               67      53     D0  00C1           MOVL    R3,SVAPTE(R7)              ;Process PTE is the Master PTE
                       D2     11  00C4           BRB     30$                        ;Join common code
                                  00C6   ;
                                  00C6   ; The loop that follows gathers pages to cluster write from the same section
                                  00C6   ; The pages must (of course) be resident, but not all of them must actually
                                  00C6   ; be modified.  For process section pages, cluster from the first page
                                  00C6   ; (guaranteed modified) through the last modified page up to the cluster size.
                                  00C6   ; For global pages, cluster write all the pages in the global writable
                                  00C6   ; section.  The state of the modified bit is indeterminate since it is
                                  00C6   ; maintained in the individual PTE's of the processes which map the section
                                  00C6   ;
             53    2C A7      C0  00C6   60$:    ADDL    INC4(R7),R3                ;Next PTE address
          04 52    16         E0  00CA           BBS     #PTE$V_TYP0,R2,70$         ;If global page (not in sh mem)
             52    28 A7      C0  00CE           ADDL    INC1(R7),R2                ;then next GPTX as well
   50    0B A7     02    00   EF  00D2   70$:    EXTZV   #0,#2,IRP$B_RMOD(R7),R0    ;Requesting mode
                       51     D4  00D8           CLRL    R1                         ;Assume no update section flags
   00D4 8F    08 A7    B1  00DA              CMPW    IRP$W_SIZE(R7),#XIP_C_LENGTH ;If extended I/O packet
                 0A    19  00E0              BLSS    75$                        ;Then
     51   00D0 C7      90  00E2              MOVB    XIP_B_UPDFLG(R7),R1        ;  Use the save update section flags
     50   00D1 C7      9A  00E7              MOVZBL  XIP_B_MAXACMODE(R7),R0     ;  Use maximized mode not requesting mode
               018E    30  00EC      75$:    BSBW    MMG$PTEPFNMFY              ;Get PFN and modify bit if resident
              10 51    E9  00EF              BLBC    R1,120$                    ;Branch if not resident
              1C A7    D6  00F2              INCL    COUNT(R7)                  ;Found another resident page
                 51    95  00F5              TSTB    R1                         ;See if it was modified
                 05    18  00F7              BGEQ    100$                       ;Branch if it was not
    0C A7     1C A7    D0  00F9      80$:    MOVL    COUNT(R7),MFYCNT(R7)       ;then update last modified page seen
           C4 18 A7    F5  00FE      100$:   SOBGTR  CLUSTER(R7),60$            ;Try the next page too
                           0102      ;
                           0102      ; Now lock all the pages in the cluster just found
                           0102      ;
              53 67    D0  0102      120$:   MOVL    SVAPTE(R7),R3              ;Get starting Master PTE
     51    0C A7 01    C3  0105              SUBL3   #1,MFYCNT(R7),R1           ;Count - 1 of pages in cluster
           51 2C A7    C4  010A              MULL    INC4(R7),R1                ;* -4 if going backwards in address space
                 12    18  010E              BGEQ    130$                       ;Branch if only 1 page or going forwards
                           0110      ;
                           0110      ; Going backwards in the address space, form the correct starting
                           0110      ; PTE addresses and virtual address.
                           0110      ;
              53 51    C0  0110              ADDL    R1,R3                      ;Form starting master PTE address
              67 53    D0  0113              MOVL    R3,SVAPTE(R7)              ;and save it
           40 A7 51    C0  0116              ADDL    R1,PROCPTE(R7)             ;Form starting process PTE address
        51 51 07       78  011A              ASHL    #7,R1,R1                   ;(count - 1) * -512
           48 A7 51    C0  011E              ADDL    R1,IRP$L_SEGVBN(R7)        ;Form starting virtual address
   18 A7    0C A7      D0  0122      130$:   MOVL    MFYCNT(R7),CLUSTER(R7)     ;Loop count is to last modified page
                           0127      ;
                           0127      ; Given the Master PTE address get each page ready for the write request
                           0127      ;
 50    83   7B800000 8F    CB  0127  150$:   BICL3   #^C<PTE$M_VALID !-         ;Get relevant bits from PTE
                           012F                      PTE$M_TYPT ! PTE$M_TYPO !-
                           012F                      PTE$M_PGFLVB>,(R3)+,R0
                 35    19  012F              BLSS    260$                       ;Branch if page is valid
                 1E    13  0131              BEQL    200$                       ;Demand zero is inconsistent
     51    50    EA 8F 78  0133              ASHL    #-PTE$V_TYPO,R0,R1         ;as would be anything other
                 17    12  0138              BNEQ    200$                       ;than transition
              03 00    EE  013A              EXTV    #PFN$V_LOC,#PFN$S_LOC,-    ;Get the page location (-4 to 3)
    52     0000'DF40       013D                      @W^PFN$AB_STATE[R0],R2
                           0142              CASE    R2,<-
                           0142                      270$,-                     ;-1 = active
                           0142                      220$,-                     ;0  = on free page list
                           0142                      220$,-                     ;1  = on modified page list
                           0142                      220$,-                     ;2  = on bad page list
                           0142                      240$ -                     ;3  = release pending
                           0142                      >,TYPE=B,LIMIT=#-1
   04'    FF 8F    52  8F  0142              CASEB   R2,#-1,S^#<<30001$-30000$>/2>-1
                           0147      30000$:
                     0044' 0147              .SIGNED_WORD     270$-30000$
                     000E' 0149              .SIGNED_WORD     220$-30000$
                     000E' 014B              .SIGNED_WORD     220$-30000$
                     000E' 014D              .SIGNED_WORD     220$-30000$
                     0015' 014F              .SIGNED_WORD     240$-30000$
                           0151      30001$:
                           0151      200$:   BUG_CHECK WRTPGSBAK,FATAL          ;Write pages back - inconsistent data base
                      FEFF 0151                      .WORD   ^XFEFF
```

```
                    0004' 0153               .IIF IDN <FATAL>,<FATAL> , .WORD        BUG$_WRTPGSBAK!4
                          0155      ;
                          0155      ; Page is on the free, modified, or bad page list, must remove it
                          0155      ;
                53     DD 0155      220$:   PUSHL   R3                              ;Save next PTE address
             FEA6'     30 0157              BSBW    MMG$REMPFN                      ;Remove page from free or modified page list
                08     BA 015A              POPR    #^M<R3>                         ;Restore next PTE address
          00    05     F0 015C      240$:   INSV    #PFN$C_WRTINPROG,#PFN$V_LOC,- ;Set state to
     0000'DF40  03        015F                      #PFN$S_LOC,@W^PFN$AB_STATE[R0] ;Write in progress
                25     11 0164              BRB     270$
                          0166
                          0166      ;
                          0166      ; Master page table entry is valid, shut off PTE copy of Modify bit, and get PFN
                          0166      ;
          51    40 A7  D0 0166      260$:   MOVL    PROCPTE(R7),R1                  ;Process page table entry address
                61     D5 016A              TSTL    (R1)                            ;See if it contains a valid PTE
                0B     18 016C              BGEQ    265$                            ;Branch if it does not
          07 61 1A     E5 016E              BBCC    #PTE$V_MODIFY,(R1),265$ ;Shut off process PTE modify bit
                          0172                                                      ;Branch if it was already off
                          0172              INVALID IRP$L_SEGVBN(R7),R1             ;Invalidate translation buffer for
          51    48 A7  D0 0172                      MOVL    IRP$L_SEGVBN(R7),R1
             3A    51  DA 0176                      MTPR    R1,S^#PR$_TBIS
                          0179                                                      ;process virtual address
                          0179
                          0179              ASSUME  PTE$V_MODIFY GE 24              ;PTE modify bit is in high byte
          FF A3    04  8A 0179      265$:   BICB    #PTE$M_MODIFY@-24,-1(R3) ;Shut off modify in master PTE
       50    50    15  00 EF 017D           EXTZV   #PTE$V_PFN,#PTE$S_PFN,R0,R0 ;Isolate PFN
     00000000'EF   50  D1 0182              CMPL    R0,MMG$GL_MAXPFN                ;Is there PFN data base? (SH MEM page)
                0C     1A 0189              BGTRU   280$                            ;Br if there is none, page is in SH MEM
     0000'DF40  80 8F  8A 018B      270$:   BICB    #PFN$M_MODIFY,@W^PFN$AB_STATE[R0] ;Page not modified
           0000'DF40   B6 0192              INCW    @W^PFN$AW_REFCNT[R0]            ;Count an I/O reference
          40 A7    04  C0 0197      280$:   ADDL    #4,PROCPTE(R7)                  ;Next process PTE address
  48 A7    00000200 8F C0 019B              ADDL    #512,IRP$L_SEGVBN(R7)           ;Next process virtual address
          80 18 A7     F5 01A3              SOBGTR  CLUSTER(R7),150$                ;Loop through each page in the cluster
                          01A7      ;
                          01A7      ; Now set up to queue the packet for writing
                          01A7      ;
          52    30 A7  D0 01A7              MOVL    BAK(R7),R2                      ;Get original backing store address
                          01AB                                                      ;section address is same for all pages
                53    67  D0 01AB           MOVL    SVAPTE(R7),R3                   ;Starting master PTE address
       50    63    15  00 EF 01AE           EXTZV   #PTE$V_PFN,#PTE$S_PFN,(R3),R0 ;Get PFN for first page to write
     00000000'EF   50  D1 01B3              CMPL    R0,MMG$GL_MAXPFN                ;Is this a shared memory gbl sec page?
                31     1A 01BA              BGTRU   320$                            ;Br if page is in shared memory gbl sec
       05 04 A7    16  E0 01BC              BBS     #PTE$V_TYP0,PTEDAT(R7),300$ ;Branch if process section page
          55    0000'CF D0 01C1             MOVL    W^MMG$GL_SYSPHD,R5              ;System header for global page
             FE37'     30 01C6      300$:   BSBW    MMG$INIBLDPKT                   ;Convert to file vbn and window
          51    0C A7  D0 01C9      310$:   MOVL    MFYCNT(R7),R1                   ;Count of pages to queue
                          01CD
                 00000002 01CD              .IF     GT,CA$_MEASURE
             0000'CF   D6 01CD              INCL    W^PMS$GL_PWRITIO                ;Count number of write I/O requests
          0000'CF  51  C0 01D1              ADDL    R1,W^PMS$GL_PWRITES             ;Count number of pages written
                          01D6              .ENDC
                          01D6
                55    57  D0 01D6           MOVL    R7,R5                           ;I/O packet address
       57    28 A5    09  78 01D9           ASHL    #9,INC1(R5),R7                  ;Restore R7
                51     DD 01DE              PUSHL   R1                              ;Save page count to return to caller
       51    51    09  9C 01E0              ROTL    #9,R1,R1                        ;Form byte count to queue
```

```
           FE19'  30  01E4          BSBW    EXE$BUILDPKTW            ;Build and queue the packet for writing
      50     01   3C  01E7          MOVZWL  #SS$_NORMAL,R0           ;Indicate packet successfully queued
             32   BA  01EA          POPR    #^M<R1,R4,R5>            ;Return byte count in R1, restore R4,R5
                  05  01EC          RSB                              ;and return
                      01ED
                      01ED  ;
                      01ED  ; COMPUTE THE VBN FOR THE FIRST PAGE IN THE CLUSTER, THE SECTION TABLE ADDRESS,
                      01ED  ; AND THE WINDOW ADDRESS.
                      01ED  ;
   55      0000'CF    D0  01ED  320$:   MOVL    W^MMG$GL_SYSPHD,R5       ;System process header (for gbl pages)
      52     52   32  01F2          CVTWL   R2,R2                    ;Section table index
51    55   20 A5  C1  01F5          ADDL3   PHD$L_PSTBASOFF(R5),R5,R1 ;Base of section table
   51      6142   DE  01FA          MOVAL   (R1)[R2],R1              ;Section table entry address
        0050 8F   BB  01FE          PUSHR   #^M<R4,R6>               ;Save registers
      56     61   D0  0202          MOVL    SEC$L_GSD(R1),R6         ;Address of Global Section Descriptor
                      0205  ;
                      0205  ; Find the relative position of this page within the section.
                      0205  ;
           FDF8'  30  0205          BSBW    MMG$FINDSHD              ;Get sh mem ctl blk & common data page
      50   10 A4  C2  0208          SUBL2   SHB$L_BASGSPFN(R4),R0    ;Get relative PFN within the sh mem
      56   54 A6  9E  020C          MOVAB   GSD$L_BASPFN1(R6),R6     ;Get adr of first PFN base in GSD
      52     04   9A  0210          MOVZBL  #GSD$C_PFNBASMAX,R2      ;Get number of PFN bases allowed
             55   D4  0213          CLRL    R5                       ;Zero relative page offset within sec
      66     50   D1  0215  330$:   CMPL    R0,(R6)                  ;Is PFN less than this base?
             09   19  0218          BLSS    340$                     ;Br if less than, not within this piece
54    66     86   C1  021A          ADDL3   (R6)+,(R6),R4            ;Get PFN past end of this piece
      54     50   D1  021E          CMPL    R0,R4                    ;Is PFN less than end of piece?
             0A   19  0221          BLSS    350$                     ;Br if less than, is within this piece
      55     86   C0  0223  340$:   ADDL2   (R6)+,R5                 ;Add pagcnt to relative page offset
      EC     52   F5  0226          SOBGTR  R2,330$                  ;Go check if PFN is in next piece
                      0229          BUG_CHECK       SCANDEADPT       ;Error, PFN must be within this GSD
            FEFF      0229                  .WORD   ^XFEFF
           0000'      022B                  .IIF DIF <CONT>,<FATAL> , .WORD BUG$_SCANDEADPT
      50     76   C2  022D  350$:   SUBL2   -(R6),R0                 ;Get relative page within this piece
      50     55   C0  0230          ADDL2   R5,R0                    ;Add page counts of other pieces to off
      50   10 A1  C0  0233          ADDL2   SEC$L_VBN(R1),R0         ;Add in base VBN
        0050 8F   BA  0237          POPR    #^M<R4,R6>               ;Restore registers
      52   0C A1  D0  023B          MOVL    SEC$L_WINDOW(R1),R2      ;Get window address
             88   11  023F          BRB     310$                     ;Join common code
```

```
         0241            .SBTTL  PTEPFNMFY - Get PFN and Modify bit from PTE
         0241
         0241     ;+
         0241     ;
         0241     ; FUNCTIONAL DESCRIPTION:
         0241     ;
         0241     ;         Return PFN and modify bit if page is a candidate for write
         0241     ; back clustering.
         0241     ;
         0241     ; CALLING SEQUENCE:
         0241     ;
         0241     ;         BSBW    MMG$PTEPFNMFY
         0241     ;
         0241     ; INPUTS:
         0241     ;
         0241     ;         R0 = Access mode to check against page owner
         0241     ;         R1 = Exclusive writer indicator
         0241     ;         R2 = Process section backing store address or GPTX
         0241     ;            = 0 if supposed to return the above or shared memory global page
         0241     ;         R3 = System Virtual Address of Page Table Entry
         0241     ;         IPL = SYNCH
         0241     ;
         0241     ; OUTPUTS:
         0241     ;
         0241     ;         R0 = Page Frame Number if successful
         0241     ;         R1 = low bit clear if page is not a candidate for write back clustering
         0241     ;              non-zero if actual error, 0 if just not a candidate
         0241     ;            = low bit set if page could be cluster written
         0241     ;              bit 7 set if modified page
         0241     ;         R2 = Process section address if process page
         0241     ;            = GPTX if global page
         0241     ;         R3   preserved
         0241     ;
         0241     ;-
         0241     ;
         0241     ; ****************************************************************************
         0241     ;
         0241     ; **************** THE FOLLOWING CODE MUST BE RESIDENT *****************
         0241     ;
     00000241             .PSECT  $MMGCOD
         0241     ;
         0241     ; ****************************************************************************
         0241     ;
```

```
                              0241        .ENABL  LSB
                              0241  ;
                              0241  ; Pages with PFN's greater than MAXPFN must be in shared memory (or  PFN-mapped,
                              0241  ; PTE$V_WINDOW set).  Shared memory pages are always mapped via global sections.
                              0241  ; There is no PFN data base for shared memory global section pages.
                              0241  ;
                              0241  SHM_PAGE:
             0053 8F    BB    0241        PUSHR   #^M<R0,R1,R4,R6>        ;Save registers
                  51    D4    0245        CLRL    R1                      ;Indicate no decrement to PTE ref count
                FDB6'   30    0247        BSBW    MMG$FINDGSDPFN          ;Find SHMGSD for this PFN
               25 50    E9    024A        BLBC    R0,30$                  ;Branch if none found (ERROR CONDITION)
      52 A6    15 A4    91    024D        CMPB    SHB$B_PORT(R4),GSD$B_CREATPORT(R6) ;Is process on creator port?
                  19    12    0252        BNEQ    20$                     ;Br if different port, cannot do update
         52    16 A6    3C    0254        MOVZWL  GSD$W_GSTX(R6),R2       ;Get global section table index
                  50    D4    0258        CLRL    R0                      ;Assume page not a wrt candidate
      13 20 A6    03    E1    025A        BBC     #SEC$V_WRT,GSD$W_FLAGS(R6),30$ ;Br if section not writeable
             0053 8F    BA    025F        POPR    #^M<R0,R1,R4,R6>        ;Restore registers
               5E 04    C0    0263        ADDL2   #4,SP                   ;Clean off saved input backing store adr
            00 52 16    E3    0266        BBCS    #PTE$V_TYPO,R2,10$      ;Treat section as a process section
                008E    31    026A  10$:  BRW     100$                    ;in WRTPGSBAK routine
      50   0384 8F      3C    026D  20$:  MOVZWL  #SS$_NOTCREATOR,R0      ;Return error code
         04 AE  50      D0    0272  30$:  MOVL    R0,4(SP)                ;Insure that error code gets to R1
             0053 8F    BA    0276        POPR    #^M<R0,R1,R4,R6>        ;Restore registers
                00D8    31    027A        BRW     180$                    ;Page not candidate for update
                              027D
                              027D  MMG$PTEPFNMFY:
                  51    DD    027D        PUSHL   R1                      ;Save exclusive writer bit
                  52    DD    027F        PUSHL   R2                      ;and the input backing store address
   51    53    15 09    EF    0281        EXTZV   #VA$V_VPN,#VA$S_VPN,R3,R1 ;Check for presence of page table
            0000'DF41   D5    0286        TSTL    @W^MMG$GL_SPTBASE[R1]   ;If SPT entry is not valid then
                  51    18    028B        BGEQ    70$                     ;this page table is not resident
   50    63    02 17    ED    028D        CMPZV   #PTE$V_OWN,#PTE$S_OWN,(R3),R0 ;Check for page owner violation
                  75    19    0292        BLSS    130$                    ;Branch if it is
  50    63   7B800000 8F CB   0294        BICL3   #^C<PTE$M_VALID !-      ;Get valid bit
                              029C                PTE$M_TYPT ! PTE$M_TYPO !- ;type bits
                              029C                PTE$M_PGFLVB>,(R3),R0  ;and PFN/GPTX from the PTE
                  72    18    029C        BGEQ    140$                    ;Branch if not valid
         3C 50    15    E0    029E        BBS     #PTE$V_WINDOW,R0,70$    ;Branch if PFN-mapped
      51    50    0D    9C    02A2  40$:  ROTL    #<32-<PTE$V_MODIFY-PFN$V_MODIFY>>,R0,R1 ;R1<7> = Modify bit
   50    50    15 00    EF    02A6        EXTZV   #PTE$V_PFN,#PTE$S_PFN,R0,R0 ;Isolate PFN
     00000000'EF  50    D1    02AB        CMPL    R0,MMG$GL_MAXPFN        ;Is this a SH MEM page?
                  8D    1A    02B2        BGTRU   SHM_PAGE                ;Br if it is a SH MEM page
      51    0000'DF40   88    02B4  50$:  BISB    @W^PFN$AB_STATE[R0],R1  ;Or in PFN copy of Modify bit
      52    0000'DF40   D0    02BA        MOVL    @W^PFN$AL_BAK[R0],R2    ;Backing store address to check
                              02C0                                        ;if page is not global
       0000'DF40  53    D1    02C0        CMPL    R3,@W^PFN$AL_PTE[R0]    ;If process PTE address is different
                  0D    13    02C6        BEQL    60$                     ;Branch if process page
  52   0000'DF40   0000'CF C3 02C8        SUBL3   W^MMG$GL_GPTBASE,@W^PFN$AL_PTE[R0],R2 ;Offset from GPT base
         52    52 1E    9C    02D1        ROTL    #<32-2>,R2,R2           ;Form Global Page Table Index
                  6E    D5    02D5  60$:  TSTL    (SP)                    ;Specified section or GPTX?
                  07    13    02D7        BEQL    80$                     ;Branch if not, return section or GPTX
               6E 52    D1    02D9        CMPL    R2,(SP)                 ;Yes, check that this one matches
                  05    13    02DC        BEQL    90$                     ;Branch if it is
                  73    11    02DE  70$:  BRB     170$                    ;Not the same, end of cluster
               6E 52    D0    02E0  80$:  MOVL    R2,(SP)                 ;Return the section or GPTX
      52    0000'DF40   D0    02E3  90$:  MOVL    @W^PFN$AL_BAK[R0],R2    ;Check that page is really writable
         66 52    16    E1    02E9        BBC     #PTE$V_TYPO,R2,170$     ;making sure it is a section,
       62 52   12  E1  02ED            BBC     #PTE$V_WRT,R2,170$          ;that it is writable
       5E 52   10  E0  02F1            BBS     #PTE$V_CRF,R2,170$          ;and that it is not copy on reference
               04  BA  02F5            POPR    #^M<R2>                     ;Fetch return section/GPTX
       07 52   16  E0  02F7            BBS     #PTE$V_TYP0,R2,110$         ;Branch if not a global page
                       02FB    ;
                       02FB    ; For the case of Global pages, the "complete" test for modified is not
                       02FB    ; possible since all process' which have valid PTE's for the global page
                       02FB    ; have their own copy of the modify bit.  This is only folded back into
                       02FB    ; the PFN data base when the page is removed from the process' working
                       02FB    ; set.  If the "exclusive write" flag is set, a Global page is only
                       02FB    ; considered modified if the process PTE or the PFN data base says that
                       02FB    ; the page is modified.  Otherwise, all Global Writable pages are considered
                       02FB    ; modified for the purposes of this write back logic.
                       02FB    ;
           04 6E       E8  02FB    100$:   BLBS    (SP),110$                   ;Branch if exclusive writer
       51     80 8F    88  02FE            BISB    #PFN$M_MODIFY,R1            ;Force modify for global writable page
          51  01       C8  0302    110$:   BISL    #1,R1                       ;Indicate successful return
          5E  04       C0  0305    120$:   ADDL    #4,SP                       ;Clean off save exclusive writer bit
              05           0308            RSB
                           0309    ;
                           0309    ; Page owner violation
                           0309    ;
     51     01EC 8F    3C  0309    130$:   MOVZWL  #SS$_PAGOWNVIO,R1           ;Return error status
                 45    11  030E            BRB     180$
                           0310    ;
                           0310    ; Page table entry was not valid, see if it is transition or global
                           0310    ;
                 41    13  0310    140$:   BEQL    170$                        ;Branch if demand zero, end of cluster
     51    50  EA 8F   78  0312            ASHL    #-PTE$V_TYP0,R0,R1          ;Transition page?
                 23    13  0317            BEQL    160$                        ;Branch if yes
                           0319    ;
                           0319    ; Process page table entry is not valid and not transition.
                           0319    ; See if it is global.
                           0319    ;
            51   01    91  0319            CMPB    #1,R1                       ;TYP1 = 0, TYP0 = 1 ?
                 35    12  031C            BNEQ    170$                        ;Branch if not global
   50    50  16  00    EF  031E            EXTZV   #PTE$V_GPTX,#PTE$S_GPTX,R0,R0 ;Isolate GPTX
                       CB  0323            BICL3   #^C<PTE$M_VALID !-          ;Get valid bit
                           0324                    PTE$M_TYP1 ! PTE$M_TYP0 !- ;type bits
                           0324                    PTE$M_PGFLVB>,-             ;and PFN/GPTX
50   0000'DF40   7B800000 8F   0324                @W^MMG$GL_GPTBASE[R0],R0 ;from the global PTE
                 05    14  032E            BGTR    150$                        ;Branch if not valid and not DZRO
                 21    13  0330            BEQL    170$                        ;Branch if demand zero to end cluster
               FF6D    31  0332            BRW     40$                         ;Process valid master PTE
     51    50  EA 8F   78  0335    150$:   ASHL    #-PTE$V_TYP0,R0,R1          ;Check for transition state
                 17    12  033A            BNEQ    170$                        ;End of cluster if not
                           033C    ;
                           033C    ; This is a transition page.  If it is on the free or modified page list
                           033C    ; or in the RELPEND or ACTIVE state, then it is still a candidate.
                           033C    ;
          03    00     EE  033C    160$:   EXTV    #PFN$V_LOC,#PFN$S_LOC,- ;Get page location (-4 to 3)
     51    0000'DF40       033F                    @W^PFN$AB_STATE[R0],R1
                           0344
                           0344            ASSUME  PFN$C_RDERR     EQ 4    ;Page read error -4
                           0344            ASSUME  PFN$C_WRTINPROG EQ 5    ;Write in progress -3
                           0344            ASSUME  PFN$C_RDINPROG  EQ 6    ;Read in progress -2
                           0344            ASSUME  PFN$C_ACTIVE    EQ 7    ;Active -1
```

```
                            0344               ASSUME  PFN$C_FREPAGLST EQ 0    ;On free page list
                            0344               ASSUME  PFN$C_MFYPAGLST EQ 1    ;On modified page list
                            0344               ASSUME  PFN$C_BADPAGLST EQ 2    ;On bad page list
                            0344               ASSUME  PFN$C_RELPEND   EQ 3    ;Release pending
                            0344
                            0344               CASE    R1,<-
                            0344                       200$,-                  ;-1 = active
                            0344                       200$,-                  ;0  = free page list
                            0344                       200$,-                  ;1  = modified page list
                            0344                       190$,-                  ;2  = bad page list
                            0344                       200$ -                  ;3  = release pending
                            0344                       >,TYPE=B,LIMIT=#-1
    04'  FF 8F     51   8F  0344               CASEB   R1,#-1,S^#<<30003$-30002$>/2>-1
                            0349       30002$:
                     0017'  0349               .SIGNED_WORD    200$-30002$
                     0017'  034B               .SIGNED_WORD    200$-30002$
                     0017'  034D               .SIGNED_WORD    200$-30002$
                     0010'  034F               .SIGNED_WORD    190$-30002$
                     0017'  0351               .SIGNED_WORD    200$-30002$
                            0353       30003$:
                            0353       ;
                            0353       ; This page is not part of the current cluster
                            0353       ;
                   51   D4  0353       170$:   CLRL    R1                      ;Return error status
                   04   BA  0355       180$:   POPR    #^M<R2>                 ;Clean off saved input backing store adr
                   AC   11  0357               BRB     120$
                            0359       ;
                            0359       ; This page is on the bad page list, if it does not have the "bad" bit
                            0359       ; set, then the page was placed there by the modified page writer due to
                            0359       ; a write error.  In this case the page should be a candidate for write back.
                            0359       ;
  F3 0000'DF40     05   E0  0359       190$:   BBS     #PFN$V_BADPAG,@W^PFN$AB_TYPE[R0],170$ ;End cluster if bad bit set
                            0360       ;
                            0360       ; This page is resident and has no I/O pending.  It may be clustered.
                            0360       ;
                   51   D4  0360       200$:   CLRL    R1                      ;No modify bit from PTE
                 FF4F   31  0362               BRW     50$
                            0365
                            0365               .DSABL  LSB
                            0365
                            0365               .END
```

```
ACB$M_QUOTA                    = 00000040            MMG$CREDEL                      ********   X   02
ACB$V_QUOTA                    = 00000006            MMG$C_LENGTH                  = FFFFFFE4
ACMODE                         = 0000000C            MMG$FINDGSDPFN                  ********   X   03
ASTADR                         = 0000001C            MMG$FINDSHD                     ********   X   03
ASTPRM                         = 00000020            MMG$GL_GPTBASE                  ********   X   03
BAK                              00000030            MMG$GL_MAXPFN                   ********   X   03
BUG$_SCANDEADPT                  ********   X   03   MMG$GL_SPTBASE                  ********   X   03
BUG$_WRTPGSBAK                   ********   X   03   MMG$GL_SYSPHD                   ********   X   03
CA$_MEASURE                    = 00000002            MMG$INADRINI                    ********   X   02
CLUSTER                          00000018            MMG$INIBLDPKT                   ********   X   03
COUNT                            0000001C            MMG$L_MAXACMODE               = FFFFFFFC
CTL$GL_PHD                       ********   X   02   MMG$L_SAVRETADR               = FFFFFFF4
DIR...                         = 00000001            MMG$L_SVSTARTVA               = FFFFFFEC
DYN$C_IRP                      = 0000000A            MMG$PTEINDX                     ********   X   03
EFN                            = 00000014            MMG$PTEPFNMFY                   0000027D R     03
EXCLWRT                          00000020            MMG$REMPFN                      ********   X   03
EXE$ALLOCBUF                     ********   X   02   MMG$RETRANGE                    ********   X   02
EXE$BUILDPKTW                    ********   X   03   MMG$UPDSECAST                   00000116 RG    02
EXE$DEANONPAGED                  ********   X   02   MMG$UPDSECPAG                   000000E9 R     02
EXE$SNGLEQUOTA                   ********   X   02   MMG$UPDSECQWT                   00000000 R     03
EXE$UPDSEC                       00000001 RG    02   MMG$WRTPGSBAK                   0000006D RG    03
FLAGS                          = 00000010            MPW$GW_MPWPFC                   ********   X   03
GSD$B_CREATPORT                = 00000052            PCB$B_PRIB                    = 0000002F
GSD$C_PFNBASMAX                = 00000004            PCB$L_PHD                     = 0000006C
GSD$L_BASPFN1                  = 00000054            PCB$L_PID                     = 00000060
GSD$W_FLAGS                    = 00000020            PCB$W_ASTCNT                  = 00000038
GSD$W_GSTX                     = 00000016            PCB$W_DIOCNT                  = 0000003E
INADR                          = 00000004            PFN$AB_STATE                    ********   X   03
INADRERR                         00000000 R     02   PFN$AB_TYPE                     ********   X   03
INC1                             00000028            PFN$AL_BAK                      ********   X   03
INC4                             0000002C            PFN$AL_PTE                      ********   X   03
IOC$DIRPOST1                     ********   X   02   PFN$AW_REFCNT                   ********   X   03
IOSB                           = 00000018            PFN$C_ACTIVE                  = 00000007
IPL$_SYNCH                     = 00000008            PFN$C_BADPAGLST               = 00000002
IRP$B_EFN                      = 00000022            PFN$C_FREPAGLST               = 00000000
IRP$B_PRI                      = 00000023            PFN$C_MFYPAGLST               = 00000001
IRP$B_RMOD                     = 0000000B            PFN$C_RDERR                   = 00000004
IRP$B_TYPE                     = 0000000A            PFN$C_RDINPROG                = 00000006
IRP$C_LENGTH                   = 000000C4            PFN$C_RELPEND                 = 00000003
IRP$L_AST                      = 00000010            PFN$C_WRTINPROG               = 00000005
IRP$L_ASTPRM                   = 00000014            PFN$M_MODIFY                  = 00000080
IRP$L_IOSB                     = 00000024            PFN$S_LOC                     = 00000003
IRP$L_IOST1                    = 00000038            PFN$V_BADPAG                  = 00000005
IRP$L_IOST2                    = 0000003C            PFN$V_LOC                     = 00000000
IRP$L_PID                      = 0000000C            PFN$V_MODIFY                  = 00000007
IRP$L_SEGVBN                   = 00000048            PHD$L_PSTBASOFF               = 00000020
IRP$W_SIZE                     = 00000008            PMS$GL_PWRITES                  ********   X   03
IRP_AST                          00000010            PMS$GL_PWRITIO                  ********   X   03
IRP_ASTPRM                       00000014            PR$_IPL                       = 00000012
IRP_EFN                          00000022            PR$_TBIS                      = 0000003A
IRP_IOSB                         00000024            PRI$_IOCOM                    = 00000001
IRP_IOST1                        00000038            PROCPTE                         00000040
IRP_LENGTH                       0000004C            PSL$S_PRVMOD                  = 00000002
IRP_PRI                          00000023            PSL$V_PRVMOD                  = 00000016
IRP_RMOD                         0000000B            PTE$M_MODIFY                  = 04000000
IRP_SEGVBN                       00000048            PTE$M_PGFLVB                  = 003FFFFF
MFYCNT                           0000000C            PTE$M_TYP0                    = 00400000
```

```
PTE$M_TYP1                       = 04000000
PTE$M_VALID                      = 80000000
PTE$S_GPTX                       = 00000016
PTE$S_OWN                        = 00000002
PTE$S_PFN                        = 00000015
PTE$V_CRF                        = 00000010
PTE$V_GPTX                       = 00000000
PTE$V_MODIFY                     = 0000001A
PTE$V_OWN                        = 00000017
PTE$V_PFN                        = 00000000
PTE$V_TYP0                       = 00000016
PTE$V_WINDOW                     = 00000015
PTE$V_WRT                        = 00000012
PTEDAT                             00000004
RETADR                           = 00000008
SAVABS...                        = 000000D4
SCH$CLREF                          ********      X     02
SCH$POSTEF                         ********      X     02
SEC$L_GSD                        = 00000000
SEC$L_VBN                        = 00000010
SEC$L_WINDOW                     = 0000000C
SEC$V_WRT                        = 00000003
SHB$B_PORT                       = 00000015
SHB$L_BASGSPFN                   = 00000010
SHM_PAGE                           00000241 R          03
SS$_ACCVIO                       = 0000000C
SS$_EXQUOTA                      = 0000001C
SS$_IVSECFLG                     = 0000016C
SS$_NORMAL                       = 00000001
SS$_NOTCREATOR                   = 00000384
SS$_NOTMODIFIED                  = 00000659
SS$_PAGOWNVIO                    = 000001EC
SVAPTE                             00000000
VA$S_VPN                         = 00000015
VA$V_VPN                         = 00000009
XIP_B_MAXACMODE                    000000D1
XIP_B_UPDFLG                       000000D0
XIP_C_LENGTH                       000000D4
XIP_L_DIREC                        000000C8
XIP_L_SCANCNT                      000000C4
XIP_L_STARTVA                      000000CC

                                              +----------------+
                                              ! Psect synopsis !
                                              +----------------+

PSECT name                       Allocation          PSECT No.  Attributes
----------                       ----------          ---------  ----------
.  ABS  .                        00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                            000000D4  (  212.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
Y$EXEPAGED                       00000189  (  393.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
$MMGCOD                          00000365  (  869.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
```

```
                                   +--------------------------+
                                   ! Performance indicators !
                                   +--------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   31     00:00:00.07    00:00:00.26
Command processing              107     00:00:00.56    00:00:01.06
Pass 1                          430     00:00:15.58    00:00:18.15
Symbol table sort                 0     00:00:02.32    00:00:02.41
Pass 2                          207     00:00:03.69    00:00:04.11
Symbol table output              19     00:00:00.15    00:00:00.15
Psect synopsis output             1     00:00:00.02    00:00:00.02
Cross-reference output            0     00:00:00.00    00:00:00.00
Assembler run totals            797     00:00:22.39    00:00:26.16

The working set limit was 1650 pages.
94749 bytes (186 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1436 non-local and 73 local symbols.
1059 source lines were read in Pass 1, producing 23 object records in Pass 2.
36 pages of virtual memory were used to define 34 macros.

                                  +----------------------------+
                                  ! Macro library statistics !
                                  +----------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                      21
_$255$DUA28:[SYSLIB]STARLET.MLB;2                   10
TOTALS (all libraries)                              31

1596 GETS were required to define 31 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SYSUPDSEC/OBJ=OBJ$:SYSUPDSEC MSRC$:SYSUPDSEC/UPDATE=(ENH$:SYSUPDSEC)+EXECML$/LIB
```

0388 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SYSSETPRA
LIS

SYSSETSSF
LIS

SYSSNDJBC
LIS

SYSSETIME
LIS

SYSSETPFM
LIS

SYSUPDSEC
LIS

SYSSNDMSG
LIS

SYSSETPRT
LIS

SYSSETPRV
LIS

SYSUNWIND
LIS

SYSSETSTK
LIS

SYSSETMOD
LIS

SYSSETEXV
LIS

SYSSETPRI
LIS

SYSSCHEVT
LIS

0389 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

WRTMFYPAG
LIS

UCBCREDEL
LIS

USRVECTOR
LIS

SYSVECTOR
LIS

SYSINIT
LIS

VERSION
LIS

SYSINI

UPCASEDAT
LIS

SYSINIT
MAP

SYSWAIT
LIS

TIMESCHDL
LIS