C 6

```
MMM         MMM     000000000     UUU         UUU  NNN         NNN  TTTTTTTTTTTTTTT
MMM         MMM     000000000     UUU         UUU  NNN         NNN  TTTTTTTTTTTTTTT
MMM         MMM     000000000     UUU         UUU  NNN         NNN  TTTTTTTTTTTTTTT
MMMMMM   MMMMMM  000         000  UUU         UUU  NNN         NNN        TTT
MMMMMM   MMMMMM  000         000  UUU         UUU  NNN         NNN        TTT
MMMMMM   MMMMMM  000         000  UUU         UUU  NNN         NNN        TTT
MMM   MMM   MMM  000         000  UUU         UUU  NNNNNN      NNN        TTT
MMM   MMM   MMM  000         000  UUU         UUU  NNNNNN      NNN        TTT
MMM   MMM   MMM  000         000  UUU         UUU  NNNNNN      NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN   NNN   NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN   NNN   NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN   NNN   NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN      NNNNNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN      NNNNNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN      NNNNNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN         NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN         NNN        TTT
MMM         MMM  000         000  UUU         UUU  NNN         NNN        TTT
MMM         MMM     000000000     UUUUUUUUUUUUUUU  NNN         NNN        TTT
MMM         MMM     000000000     UUUUUUUUUUUUUUU  NNN         NNN        TTT
MMM         MMM     000000000     UUUUUUUUUUUUUUU  NNN         NNN        TTT
```

FILEID**STACP

```
  SSSSSSSS  TTTTTTTTTT    AAAAAA      CCCCCCCC  PPPPPPPP
  SSSSSSSS  TTTTTTTTTT    AAAAAA      CCCCCCCC  PPPPPPPP
SS              TT      AA      AA  CC          PP      PP
SS              TT      AA      AA  CC          PP      PP
SS              TT      AA      AA  CC          PP      PP
SS              TT      AA      AA  CC          PP      PP
  SSSSSS        TT      AA      AA  CC          PPPPPPPP
  SSSSSS        TT      AA      AA  CC          PPPPPPPP
        SS      TT      AAAAAAAAAA  CC          PP
        SS      TT      AAAAAAAAAA  CC          PP
        SS      TT      AA      AA  CC          PP              ....
        SS      TT      AA      AA  CC          PP              ....
SSSSSSSS        TT      AA      AA    CCCCCCCC  PP              ....
SSSSSSSS        TT      AA      AA    CCCCCCCC  PP              ....

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 STACP (
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-001'
0                       ) =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  MOUNT Utility Structure Level 1
1 !
1 ! ABSTRACT:
1 !
1 !       This routine hooks up the VCB to the UCB being mounted, finds or
1 !       creates the AQB and starts up the ACP.
1 !
1 ! ENVIRONMENT:
1 !
1 !       STARLET operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1 !
1 !
1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  19-Oct-1977  15:43
1 !
1 ! MODIFIED BY:
1 !
1 !       V04-001 HH0058          Hai Huang               13-Sep-1984
1 !               Properly mark up the UCB if the MOUNT QIO fails. This
1 !               will cause $DALLOC_DEVS to properly handle the device
1 !               lock.
1 !
1 !       V03-021 HH0045          Hai Huang               10-Aug-1984
```

```
1 !                       Prevent race condition between mount and dismount
1 !                       (i.e. HH0025) for Files-11 level 1 volumes.
1 !
1 !               V03-020 CDS0009         Christian D. Saether    25-Jul-1984
1 !                       Fix name of sysgen parameter for fourth pool.
1 !
1 !               V03-019 HH0041          Hai Huang               24-Jul-1984
1 !                       Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
1 !
1 !               V03-018 CDS0008         Christian D. Saether    15-July-1984
1 !                       Raise minimum requirement for header cache to 3 buffers.
1 !                       Add fourth pool to buffer cache for directory index.
1 !
1 !               V03-017 HH0029          Hai Huang               29-Jun-1984
1 !                       Enhance the block cache algorithm to retry with a reduced
1 !                       cache size if the initial cache allocation fails.
1 !
1 !               V03-016 HH0025          Hai Huang               21-Jun-1984
1 !                       Prevent the race condition between mount and dismount
1 !                       by "deallocating" the device as for non-private mounts
1 !                       before setting device-mounted bit.
1 !
1 !               V03-015 HH0004          Hai Huang               13-Mar-1984
1 !                       Fix truncation errors introduced by cluster-wide
1 !                       mount support.
1 !
1 !               V03-014 CDS0007         Christian D. Saether     3-Mar-1984
1 !                       Set CLF_UNLOCKDB prior to allocating buffer cache.
1 !
1 !               V03-013 CDS0006         Christian D. Saether    28-Feb-1984
1 !                       Store F11BC$T_CACHENAME.  Restrict size of block
1 !                       cache to 2^15-2 buffers total.
1 !
1 !               V03-012 CDS0005         Christian D. Saether     6-Feb-1984
1 !                       Add support for xqp block caches.
1 !
1 !               V03-011 CDS0004         Christian D. Saether    17-Aug-1983
1 !                       Change references to XQP to STORED_CONTEXT [XQP].
1 !
1 !               V03-010 RAS0180         Ron Schaefer             4-Aug-1983
1 !                       Fix broken calls caused by changes in DISMNTSHR.
1 !
1 !               V03-009 STJ56368        Steven T. Jeffreys,     27-May-1983
1 !                       Fix /PROCESSOR=SAME:<junk> bug.  Return NOACPDEV to force
1 !                       mount to abort rather than crash later on.
1 !
1 !               V03-008 STJ50311        Steven T. Jeffreys,     22-Feb-1983
1 !                       Explicitly assign a channel to the first volume of a
1 !                       tape volume set.
1 !
1 !               V03-007 CWH1002         CW Hobbs                19-Feb-1983
1 !                       Change $CREPRC call so that we change the extended pid returned
1 !                       by $CREPRC to the internal pid needed in the AQB.  Also change
1 !                       $DELPRC and $WAKE to use the new pid.  Create a new local ACP_PID
1 !                       which contains the EPID.
1 !
1 !               V03-006 CDS0003         Christian D. Saether    12-Jan-1983
```

```
1 !             Always flag the AQB for the xqp as unique so non-xqp
1 !             mounts don't stumble onto it and use the xqp when
1 !             an acp is desired.
1 !
1 !     V03-005 CDS0002         Christian D. Saether     6-Jan-1983
1 !             The EXP$XQP translation is done in MOUNT_VOLUME now.
1 !
1 !     V03-004 CDS0001         C Saether                18-Jul-1982
1 !             - Put hooks in to allow initial testing of procedure based
1 !             file system.  To wit, if ods-2 and EXP$XQP translates,
1 !             then ignore acp options such as unique, etc., and leave
1 !             an AQB around with an ACPPID field of 0 to trigger a
1 !             different queueing mechanism in qio.  Don't create an
1 !             ACP either in that case.
1 !
1 !     V03-003 ACG47812        Andrew C. Goldstein,     20-Jul-1982  18:52
1 !             Supply explicit quotas in creating ACP mailbox
1 !
1 !     V03-002 STJ0313         Steven T. Jeffreys,      02-Jul-1982
1 !             Fix ACP process quota list bug.
1 !
1 !     V03-001 STJ0253         Steven T. Jeffreys,      03-Apr-1982
1 !             - Use the system process quota list when creating an ACP.
1 !             - Use common I/O routines where possible, and make the
1 !               use of event flags more robust.
1 !             - Add ACP type code to ACP process name.
1 !
1 !     V02-004 STJ0191         Steven T. Jeffreys,      02-Feb-1982
1 !             Zero GLOBAL and OWN storage to guaranty restartablity.
1 !
1 !     V02-003 DMW0007         David Micahel Walp       10-Jun-1981
1 !             Use FIRST_CHANNEL instead of CHANNEL, for tapes in the
1 !             MOUNT QIOW.
1 !
1 !     V02-002 DMW0005         David Michael Walp       20-May-1981
1 !             Changed debugging terminal from _TTB1: to ACP$DEBUG_TERMINAL
1 !
1 !     V02-001 ACG0167         Andrew C. Goldstein,     18-Apr-1980  13:39
1 !             Previous revision history moved to MOUNT.REV
1 !**
1
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 REQUIRE 'SRC$:MOUDEF.B32';
1
1 FORWARD ROUTINE
1       SETUP_BLOCKCACHE,
1       ALLOCATE_PAGED;
```

```
1 GLOBAL ROUTINE START_ACP (UCB, VCB, TYPE) : NOVALUE =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine hooks up the VCB to the UCB being mounted, finds or
1 !       creates the AQB and starts up the ACP.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !       START_ACP (ARG1, ARG2, ARG3)
1 !
1 ! INPUT PARAMETERS:
1 !       ARG1: address of UCB
1 !       ARG2: address of VCB
1 !       ARG3: type code of ACP wanted
1 !
1 ! IMPLICIT INPUTS:
1 !       MOUNT parser database
1 !       CHANNEL: I/O channel assigned to device being mounted
1 !
1 ! OUTPUT PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !       device characteristics altered, VCB hooked up to UCB, ACP started
1 !
1 !--
1
2 BEGIN
2
2 MAP
2       UCB             : REF BBLOCK,   ! address of UCB being mounted
2       VCB             : REF BBLOCK,   ! address of VCB being mounted
2       TYPE            : BYTE;         ! type code of desired ACP
2
2 LINKAGE
2       L_CVT_DEVNAM    = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 5,
2                               REGISTER = 4; REGISTER = 1) :
2                       NOTUSED (6,7,8,9,10,11),
2
2       IOC_SEARCH      = JSB (REGISTER = 4) :
2                       NOPRESERVE (2, 3, 5)
2                       GLOBAL (ACP_DEVICE = 1);
2
2 LABEL
2       ACP_SEARCH;                             ! main ACP search loop
2
2 LITERAL
2       TIMER_EFN       = MOUNT_EFN + 1,! EFN for timer request
```

```
                PQL_LENGTH      = 100,              ! Size of local PQL length
                                                    !  Note: PQL_LENGTH >= PQL$C_SYSPQLLEN+6
                HALF_SECOND     = 5000000,          ! 1/2 second in 100 nsec units
                ACP_UIC         = 1^16 + 3,         ! UIC to run ACP's under [1,3]
                MAILBOX_CHARLEN = 16,               ! length of mailbox characteristics buffer
                TERM_BUFFER_LEN = 8,                ! length of termination message buffer
                PROCBUF_LEN     = 16,               ! maximum length of ACP process name
                FILEBUF_LEN     = 64;               ! maximum length of ACP file name

        LOCAL
                FIRST_CHANNEL,                      ! channel assigned to 1st tape drive
                CREATE_ACP,                         ! ACP creation flag
                STATUS,                             ! general service status value
                CLASS           : BYTE,             ! device class code for ACP sharing
                FILE_PREF       : REF VECTOR [,BYTE], ! address of ACP prefix string
                FILE_PREF_LEN,                      ! length of ACP prefix string
                SWAP_FLAG,                          ! ACP swap status flag
                WORKING_SET,                        ! working set quota for ACP
                ACP_PID,                            ! local extended pid for created ACP process
                ACP_TYPE        : VECTOR [2, BYTE], ! type string for ACP proc name
                ACP_UCB         : REF BBLOCK,       ! UCB of ACP to be used
                RVT             : REF BBLOCK,       ! address of disk RVT if multivolume
                AQB             : REF BBLOCK,       ! address of AQB to be used
                P               : REF BBLOCK,       ! pointer to chase AQB list
                QUOTA_LIST      : VECTOR [PQL_LENGTH, BYTE]; ! ACP quota list

        OWN
                OWN_START       : VECTOR [0]    ADDRESSING_MODE (GENERAL),
                                                    ! Mark start of OWN storage
                                                    ! ACP termination buffer
                MAILBOX_CHAR    : BBLOCK [MAILBOX_CHARLEN] ADDRESSING_MODE (GENERAL),
                                                    ! mailbox characteristics buffer
                MAILBOX_DESC    : VECTOR [2]    ADDRESSING_MODE (GENERAL),
                                                    ! mailbox characteristics buffer descriptor
                TERM_BUFFER     : VECTOR [TERM_BUFFER_LEN, BYTE] ADDRESSING_MODE (GENERAL),
                                                    ! ACP termination message buffer
                PROC_NAME       : VECTOR [2]    ADDRESSING_MODE (GENERAL),
                                                    ! string descriptor of ACP process name
                PROCBUF         : VECTOR [PROCBUF_LEN, BYTE] ADDRESSING_MODE (GENERAL),
                                                    ! string buffer for ACP process name
                FILE_NAME       : VECTOR [2]    ADDRESSING_MODE (GENERAL),
                                                    ! string descriptor of ACP file name
                FILEBUF         : VECTOR [FILEBUF_LEN, BYTE] ADDRESSING_MODE (GENERAL),
                                                    ! string buffer for ACP file name
                IO_STATUS       : VECTOR [2]    ADDRESSING_MODE (GENERAL),
                                                    ! I/O status block

                ! the following is a descriptor for the terminal used
                !  when debugging the ACP
                !
                TRANSBUF        : VECTOR [16, BYTE] ADDRESSING_MODE (GENERAL),
                TRANSDESC       : VECTOR [ 2, LONG] ADDRESSING_MODE (GENERAL),
                OWN_END         : VECTOR [0] ADDRESSING_MODE (GENERAL);
                                                    ! Mark end of OWN storage

        LITERAL
                OWN_LENGTH      = OWN_END - OWN_START;
```

```
EXTERNAL LITERAL
        PQL$C_SYSPQLLEN;                        ! System process quota list length
                                                ! (without the list terminator)
EXTERNAL
        CACHE_STATUS,                           ! status of block cache allocation
        STORED_CONTEXT  : BITVECTOR,
        PHYS_NAME       : VECTOR VOLATILE,
                                                ! vector of physical device descriptors
        DEVICE_CHAR     : BBLOCK,               ! DEVICE CHARACTERISTICS OF DEVICE BEING MOUNTED
        MOUNT_OPTIONS   : BITVECTOR,            ! command options
        CLEANUP_FLAGS   : BITVECTOR,            ! cleanup action flags
        CHANNEL,                                ! channel assigned to device
        MAILBOX_CHANNEL,                        ! channel number assigned to mailbox
        HOME_BLOCK      : BBLOCK,               ! address of volume home block if disk
        ACP_STRING      : VECTOR,               ! string descriptor of ACP device or name
        REAL_RVT        : REF BBLOCK,           ! address of RVT used, if any
        REAL_AQB        : REF BBLOCK,           ! address of AQB allocated
        REAL_WCB        : REF BBLOCK,           ! address of index file window
        PQL$AB_SYSPQL   : BBLOCK ADDRESSING_MODE (GENERAL),
                                                ! system process quota list
        CTL$GL_PCB      : REF BBLOCK ADDRESSING_MODE (GENERAL),
                                                ! address of our PCB
        IOC$GL_AQBLIST  : REF BBLOCK ADDRESSING_MODE (ABSOLUTE),
                                                ! system AQB listhead
        EXE$GL_FLAGS    : BITVECTOR ADDRESSING_MODE (ABSOLUTE),
                                                ! system flags longword
        ACP$GW_WORKSET  : WORD ADDRESSING_MODE (ABSOLUTE),
                                                ! disk ACP working set
        ACP$GW_MAPCACHE : WORD ADDRESSING_MODE (ABSOLUTE),
                                                ! disk ACP bitmap cache size
        ACP$GW_HDRCACHE : WORD ADDRESSING_MODE (ABSOLUTE),
                                                ! disk ACP header cache size
        ACP$GW_DIRCACHE : WORD ADDRESSING_MODE (ABSOLUTE),
                                                ! disk ACP directory cache size
        ACP$GB_SWAPFLGS : BITVECTOR ADDRESSING_MODE (ABSOLUTE),
                                                ! ACP swap mode flags
        ACP$GB_BASEPRIO : BYTE ADDRESSING_MODE (ABSOLUTE);
                                                ! ACP base priority

EXTERNAL LITERAL
        EXE$V_MULTACP   : UNSIGNED (6),         ! multiple ACP bit in system flags
        EXE$V_INIT      : UNSIGNED (6),         ! ACP initialized bit in system flags
        ACP$V_SWAPSYS   : UNSIGNED (6),         ! swap /SYSTEM (etc.) ACP's
        ACP$V_SWAPGRP   : UNSIGNED (6),         ! swap /GROUP ACP's
        ACP$V_SWAPPRV   : UNSIGNED (6),         ! swap private ACP's
        ACP$V_SWAPMAG   : UNSIGNED (6);         ! swap magtape ACP's

EXTERNAL ROUTINE
        LOCK_IODB       : ADDRESSING_MODE (LONG_RELATIVE),
                                                ! lock I/O database mutex
        UNLOCK_IODB     : ADDRESSING_MODE (LONG_RELATIVE),
                                                ! unlock the above
        ALLOCATE_MEM,                           ! allocate system dynamic memory
        DEALLOCATE_MEM,                         ! deallocate same
        SET_DATACHECK,                          ! set data check attributes for volume
```

```
2                IOC$CVT_DEVNAM   : L_CVT_DEVNAM ADDRESSING_MODE (ABSOLUTE),
2                                                ! get fully expanded device name
2                IOC$SEARCHDEV    : IOC_SEARCH ADDRESSING_MODE (ABSOLUTE);
2                                                ! search I/O database for device
2
2
2   ! Intialize the OWN storage.  Most of it will be zeroed, but some
2   ! locations must be set nonzero.
2   !
2   CH$FILL (0, OWN_LENGTH, OWN_START);
2   MAILBOX_DESC [0] = MAILBOX_CHARLEN;
2   MAILBOX_DESC [1] = MAILBOX_CHAR;
2   TRANSDESC [0] = 16;
2   TRANSDESC [1] = TRANSBUF;
2   CACHE_STATUS  = 1;
2
2   !
2   ! Establish whether we are creating an ACP. This is controlled by
2   ! the /PROCESSOR qualifier; the default is controlled by EXE$V_MULTACP
2   ! in the system mask: either a common ACP for each type, or one ACP per
2   ! device class per ACP type.
2   !
2   ! An outer loop exists around this code to handle the situation where two
2   ! users attempt to create the same common ACP (detected by seeing the
2   ! AQB$V_CREATING bit in the found AQB). When this happens we simply wait
2   ! a while and try all over. If the condition sticks for 30 seconds we
2   ! give up on grounds of a sick I/O database.
2   !
2
2   UCB[UCB$L_VCB] = .VCB;                      ! set up VCB pointer in UCB
2
3   ACP_SEARCH: BEGIN
3   DECR J FROM 60 TO 1 DO
4   BEGIN
4
4   LOCK_IODB ();
4
4   CREATE_ACP = 0;                             ! assume no ACP creation
4
4   CLASS = .BBLOCK[.UCB[UCB$L_DDB], DDB$B_ACPCLASS];
4
4   ! If this volume is part of a disk volume set and another volume is already
4   ! mounted, use the ACP of that volume.
4   !
4
4   IF NOT .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_SQD]
4   AND .REAL_RVT NEQ 0 AND .REAL_RVT[RVT$W_REFC] NEQ 1
4   THEN
5       BEGIN
6       IF (
6           INCR J FROM 1 TO .REAL_RVT[RVT$B_NVOLS]
6           DO
7               BEGIN
7               ACP_UCB = .VECTOR [REAL_RVT[RVT$L_UCBLST], .J-1];
7               IF .ACP_UCB NEQ 0 AND .ACP_UCB NEQ .UCB
7               THEN EXITLOOP 0;
7               END
```

```
6           )
5       THEN BUG_CHECK (NOTUCBRVT, FATAL, 'Failed to find UCB pointer in RVT');
5
5       IF NOT .BBLOCK [ACP_UCB[UCB$L_DEVCHAR], DEV$V_MNT]
5       THEN BUG_CHECK (NOTUCBRVT, FATAL, 'Bad UCB pointer in RVT');
5
5       AQB = .BBLOCK [.ACP_UCB[UCB$L_VCB], VCB$L_AQB];
5       IF .AQB[AQB$B_ACPTYPE] NEQ .TYPE
5       THEN BUG_CHECK (NOTUCBRVT, FATAL, 'Bad UCB pointer in RVT');
5       END
5
5   ! If a unique ACP is explicitly requested, set to create one.
5   !
5
4   ELSE IF .MOUNT_OPTIONS[OPT_UNIQUEACP] OR .MOUNT_OPTIONS[OPT_FILEACP]
4   THEN CREATE_ACP = 1
4
4   ! If the SAME qualifier was specified, find the device given and make sure
4   ! it in fact has an ACP.
4   !
4
4   ELSE IF .MOUNT_OPTIONS[OPT_SAMEACP]
4   THEN
5       BEGIN
6           BEGIN
6           GLOBAL REGISTER ACP_DEVICE = 1;
6           ACP_DEVICE = ACP_STRING[0];
6           STATUS = IOC$SEARCHDEV (.CTL$GL_PCB);
6           ACP_UCB = .ACP_DEVICE;
5           END;
5       IF NOT .STATUS
5       THEN
6           BEGIN
6           UNLOCK_IODB ();
6           ERR_EXIT (MOUN$_NOACPDEV);
5           END;
5       IF NOT .BBLOCK [ACP_UCB[UCB$L_DEVCHAR], DEV$V_MNT]
5       OR NOT .BBLOCK [ACP_UCB[UCB$L_DEVCHAR], DEV$V_FOD]
5       OR .BBLOCK [ACP_UCB[UCB$L_DEVCHAR], DEV$V_FOR]
5       THEN
6           BEGIN
6           UNLOCK_IODB ();
6           ERR_EXIT (MOUN$_NOACPDEV);
5           END;
5       AQB = .BBLOCK [.ACP_UCB[UCB$L_VCB], VCB$L_AQB];
5       IF .AQB[AQB$B_ACPTYPE] NEQ .TYPE
5       THEN
6           BEGIN
6           UNLOCK_IODB ();
6           ERR_EXIT (MOUN$_INCOMPACP);
5           END;
5       END
5
5   ! Otherwise we use the default ACP (one ACP per class/type). Search the
5   ! system AQB list for a suitable AQB (of the right type and marked system
5   ! default or right type and class and marked class default).
5   !
```

```
; 5
; 4 ELSE
; 5     BEGIN
; 5     AQB = .IOC$GL_AQBLIST;
; 5     UNTIL .AQB EQL 0 DO
; 6         BEGIN
; 6         IF .EXE$GL_FLAGS[EXE$V_MULTACP]
; 6         THEN
; 7             BEGIN
; 7             IF .AQB[AQB$V_DEFCLASS]
; 7             AND .AQB[AQB$B_ACPTYPE] EQL .TYPE
; 7             AND .AQB[AQB$B_CLASS] EQL .CLASS
; 7             THEN
; 7                 EXITLOOP;
; 7             END
; 6         ELSE
; 7             BEGIN
; 7             IF .AQB[AQB$V_DEFSYS]
; 7             AND .AQB[AQB$B_ACPTYPE] EQL .TYPE
; 7             THEN
; 7                 EXITLOOP;
; 6             END;
; 6         AQB = .AQB[AQB$L_LINK];
; 5         END;
; 5     IF .AQB EQL 0 THEN CREATE_ACP = 1;
; 4     END;
; 4
; 4 ! If we are creating an ACP, now allocate the AQB thereto. Fill in the
; 4 ! AQB and hook it into the system AQB list. Note that this must be done under
; 4 ! the I/O database lock since the list is singly linked.
; 4 !
; 4
; 4 IF .CREATE_ACP
; 4 THEN
; 5     BEGIN
; 5     AQB = ALLOCATE_MEM (AQB$C_LENGTH, 0);
; 5     AQB[AQB$B_TYPE] = DYN$C_AQB;
; 5     AQB[AQB$B_MNTCNT] = 1;
; 5     AQB[AQB$B_ACPTYPE] = .TYPE;
; 5
; 5     IF .STORED_CONTEXT [XQP]
; 5     THEN
; 5         AQB [AQB$V_XQIOPROC] = 1
; 5     ELSE
; 5
; 5 ! Set the CREATING flag to interlock against multiple mounts trying
; 5 ! to create the same ACP.  This interlock is not necessary when
; 5 ! creating an AQB for an XQP file system because, for xqps, the IO database
; 5 ! mutex is held until the AQB is completely initialized, whereas with
; 5 ! an ACP, the mutex is released while the ACP process is being created.
; 5 !
; 5
; 5         AQB[AQB$V_CREATING] = 1;
; 5
; 5     IF .MOUNT_OPTIONS[OPT_UNIQUEACP] OR .MOUNT_OPTIONS[OPT_FILEACP]
; 5     THEN
; 5         AQB[AQB$V_UNIQUE] = 1
```

```
5         ELSE IF .EXE$GL_FLAGS[EXE$V_MULTACP]
5         THEN
6             BEGIN
6             AQB[AQB$V_DEFCLASS] = 1;
6             AQB[AQB$B_CLASS] = .CLASS;
6             END
5         ELSE
5             AQB[AQB$V_DEFSYS] = 1;
5
5         AQB[AQB$L_ACPQFL] = AQB[AQB$L_ACPQFL];
5         AQB[AQB$L_ACPQBL] = AQB[AQB$L_ACPQFL];
5         AQB[AQB$L_LINK] = .IOC$GL_AQBLIST;
5         IOC$GL_AQBLIST = .AQB;
5         REAL_AQB = .AQB;
5         CLEANUP_FLAGS[CLF_DELAQB] = 1;
5         LEAVE ACP_SEARCH;
5         END
5
5     ! If we are not creating a new ACP then we have found an AQB to use. Check
5     ! that it is not in transition. If OK, bump its mount count. Otherwise,
5     ! release the I/O database mutex, wait a while, and try all over.
5     !
5
4     ELSE
5         BEGIN
5         IF NOT .AQB[AQB$V_CREATING]
5         THEN
6             BEGIN
6             AQB[AQB$B_MNTCNT] = .AQB[AQB$B_MNTCNT] + 1;
6             LEAVE ACP_SEARCH;
5             END;
4         END;
4
4     UNLOCK_IODB ();
4
4     ! The AQB we want to use is in transition (i.e., it is a new ACP being created).
4     ! Since this may or may not be successful, we cannot use it at this time.
4     ! Time out and retry.
4     !
4
5     IF $SETIMR (REQIDT = 999, EFN = TIMER_EFN, DAYTIM = UPLIT (-HALF_SECOND, -1))
5     THEN
5         BEGIN
5         $WAITFR (EFN = TIMER_EFN);
5         $CANTIM (REQIDT = 999);
5         $SETEF  (EFN = TIMER_EFN);
4         END;
4
3     END;                                      ! end of ACP search retry loop
3
3     ! If we fall out of the loop, the AQB we want has been in transition for
3     ! 10 seconds and is clearly in trouble.
3     !
3
3     ERR_EXIT (MOUN$_AQBTIME);
3
2     END;                                      ! end of block ACP_SEARCH
```

```

IF NOT .CREATE_ACP
THEN
    UNLOCK_IODB ()
ELSE IF .AQB [AQB$V_XQIOPROC]
THEN
    BEGIN
    LOCAL
        CACHE_HDR           : REF BBLOCK,
        LENGTH;

    CLEANUP_FLAGS [CLF_UNLOCKDB] = 1;

    AQB [AQB$L_BUFCACHE] = SETUP_BLOCKCACHE (CACHE_STATUS);

    AQB [AQB$V_CREATING] = 0;
    CLEANUP_FLAGS [CLF_DELAQB] = 0;
    CLEANUP_FLAGS [CLF_UNLOCKDB] = 0;

    CACHE_HDR = .AQB [AQB$L_BUFCACHE];

    IOC$CVT_DEVNAM (15, CACHE_HDR [F11BC$T_CACHENAME] + 1, .UCB, 0; LENGTH);

    (CACHE_HDR [F11BC$T_CACHENAME])<0,8> = .LENGTH + 8;
    CH$MOVE (8, UPLIT BYTE ('XQPCACHE'),
                CACHE_HDR [F11BC$T_CACHENAME] + 1 + .LENGTH);

    UNLOCK_IODB ();

    END
ELSE

! Now create the ACP process if needed. Construct the process name from
! the device name and unit in the form ddcuACP; construct the
! ACP file name from the supplied name or the default name in the DDB.
! Note that the ACP file prefix depends upon whether or not a file system
! exists for the system disk. Normally, we use 'SYS$SYSTEM' but when this
! is the mount of the system disk, we use '[SYSEXE]' since FILEREAD doesn't
! do logical name translation.
!

    BEGIN

    UNLOCK_IODB ();

    PROC_NAME[0] = PROCBUF_LEN;
    PROC_NAME[1] = PROCBUF;
    ACP_TYPE[0] = 1;
    ACP_TYPE[1] = 'A' - 1 + .TYPE;

    $FAO (
        DESCRIPTOR ('!AC!UW!AC!AC'),
        PROC_NAME[0],
        PROC_NAME[0],
        BBLOCK [.UCB[UCB$L_DDB], DDB$T_NAME],
        .UCB[UCB$W_UNIT],
        ACP_TYPE,
```

```
            UPLIT BYTE (3, 'ACP')
            );

        IF .EXE$GL_FLAGS[EXE$V_INIT]
        THEN
            BEGIN
            FILE_PREF_LEN = %CHARCOUNT ('SYS$SYSTEM:');
            FILE_PREF = UPLIT BYTE ('SYS$SYSTEM:');
            END
        ELSE
            BEGIN
            FILE_PREF_LEN = %CHARCOUNT ('[SYSEXE]');
            FILE_PREF = UPLIT BYTE ('[SYSEXE]');
            END;

        FILE_NAME[1] = FILEBUF;
        IF .MOUNT_OPTIONS[OPT_FILEACP]
        THEN
            BEGIN
            FILE_NAME[0] = .ACP_STRING[0] + .FILE_PREF_LEN;
            CH$COPY (.FILE_PREF_LEN, .FILE_PREF, .ACP_STRING[0], .ACP_STRING[1],
                     0, .FILE_NAME[0], FILEBUF);
            END
        ELSE
            BEGIN
            FILE_NAME[0] = .FILE_PREF_LEN + 11;
            CH$COPY (.FILE_PREF_LEN, .FILE_PREF, 3, BBLOCK [.UCB[UCB$L_DDB], DDB$L_ACPD],
                     8, UPLIT BYTE ('AACP.EXE'),
                     0, .FILE_PREF_LEN+11, FILEBUF);
            IF .MOUNT_OPTIONS[OPT_IS_FILES11B]
            THEN (FILEBUF+.FILE_PREF_LEN+3)<0,8> = 'B';
            END;

    ! Create a mailbox which will receive the termination message in case the
    ! ACP bombs. The $GETCHN call is used to obtain the mailbox unit number.
    !

        MAILBOX_CHANNEL = 0;
        STATUS = $CREMBX (CHAN   = MAILBOX_CHANNEL,
                          MAXMSG = 132,
                          BUFQUO = 132,
                          PROMSK = %X'FFED');          ! system writable, owner readable
        IF NOT .STATUS THEN ERR_EXIT (.STATUS);
        CLEANUP_FLAGS[CLF_DEASSMBX] = 1;

        $GETCHN (CHAN   = .MAILBOX_CHANNEL,
                 PRIBUF = MAILBOX_DESC[0]);

        SWAP_FLAG = PRC$M_PSWAPM;
        IF .ACP$GB_SWAPFLGS[ (IF .DEVICE_CHAR[DEV$V_SQD] THEN ACP$V_SWAPMAG
                             ELSE IF .AQB[AQB$V_DEFSYS] OR .AQB[AQB$V_DEFCLASS]
                                     OR .MOUNT_OPTIONS[OPT_SYSTEM] THEN ACP$V_SWAPSYS
                             ELSE IF .MOUNT_OPTIONS[OPT_GROUP] THEN ACP$V_SWAPGRP
                             ELSE ACP$V_SWAPPRV
                             )]
        THEN SWAP_FLAG = 0;
```

```
! Compute the working set quota for the ACP
!

    IF .DEVICE_CHAR[DEV$V_SQD]
    THEN
        WORKING_SET = 128                   ! working set for magtape ACP
    ELSE
        BEGIN
        WORKING_SET = .ACP$GW_WORKSET;  ! working set for disk ACP
        IF .WORKING_SET EQL 0
        THEN WORKING_SET = .ACP$GW_MAPCACHE + .ACP$GW_HDRCACHE + .ACP$GW_DIRCACHE + 150;
        END;

    ! Make a local copy of the system quota list and append the calculated
    ! ACP working set quota to the end of the list.  As the system quota list
    ! may grow, take care to leave room for our custom quota entry.
    !

    CH$MOVE (MIN (PQL$C_SYSPQLLEN,(PQL_LENGTH-6)), PQL$AB_SYSPQL, QUOTA_LIST);
    (QUOTA_LIST+PQL$C_SYSPQLLEN)<0,8> = PQL$_WSQUOTA;
    (QUOTA_LIST+PQL$C_SYSPQLLEN+1)<0,32> = .WORKING_SET;
    (QUOTA_LIST+PQL$C_SYSPQLLEN+5)<0,8> = PQL$_LISTEND;

    ! for debugging the ACP
    !
P   $TRNLOG ( LOGNAM = DESCRIPTOR ('ACP$DEBUG_TERMINAL'),
P             RSLLEN = TRANSDESC,
              RSLBUF = TRANSDESC );

P   STATUS = $CREPRC (
P                       PIDADR = ACP_PID,              ! get extended pid in local
P                       IMAGE  = FILE_NAME[0],
P                       PRCNAM = PROC_NAME[0],
P                       QUOTA  = QUOTA_LIST,
P                       PRVADR = UPLIT (-1,-1),
P                       BASPRI = .ACP$GB_BASEPRIO,
P                       UIC    = ACP_UIC,
P                       MBXUNT = .MAILBOX_CHAR[DIB$W_UNIT],
P                       STSFLG = (.SWAP_FLAG OR PRC$M_HIBER),       ! hibernate immediately
                        INPUT  = TRANSDESC );          ! DEBUG the ACP

    IF NOT .STATUS THEN ERR_EXIT (.STATUS);

! We created the process in the hibernate state because we need to twiddle the PID before it gets put in the
! $CREPRC has put an extended PID in ACP_PID, we need to convert the EPID to an internal PID for the AQB.  W
! this is done we can wake the ACP.
!
    BEGIN           ! put a block around the routine declarations
    LINKAGE
        cvt_lnk = JSB (REGISTER=0) : PRESERVE (1,2,3,4,5) NOTUSED (6,7,8,9,10,11);
    EXTERNAL ROUTINE
        EXE$EPID_TO_IPID : cvt_lnk ADDRESSING_MODE (GENERAL);
    AQB[AQB$L_ACPPID] = EXE$EPID_TO_IPID (.ACP_PID);
    END;
    STATUS = $WAKE (PIDADR = ACP_PID);              ! wake it up using the epid
    IF NOT .STATUS THEN ERR_EXIT (.STATUS);
```

```
! Now wait for the ACP to come up. We do this by sitting in a time out loop
! waiting for the transition bit in the AQB to clear, and also waiting for
! a message on the mailbox in case the ACP dies.
!

    $QIO (
        CHAN    = .MAILBOX_CHANNEL,
        FUNC    = IO$_READVBLK,
        EFN     = MOUNT_EFN,
        IOSB    = IO_STATUS[0],
        P1      = TERM_BUFFER,
        P2      = TERM_BUFFER_LEN
        );
    $SETEF (EFN = MOUNT_EFN);

    IF DECR J FROM 60 TO 1 DO
        BEGIN
        IF $SETIMR (REQIDT = 999, EFN = TIMER_EFN, DAYTIM = UPLIT (-HALF_SECOND, -1))
        THEN
            BEGIN
            $WAITFR (EFN = TIMER_EFN);
            $CANTIM (REQIDT = 999);
            $SETEF (EFN = TIMER_EFN);
            END;

        IF NOT .AQB[AQB$V_CREATING]
        THEN EXITLOOP 0;

        IF .IO_STATUS[0] NEQ 0
        THEN
            BEGIN
            IF NOT .IO_STATUS[0] THEN ERR_EXIT (.IO_STATUS[0]);
            IF .TERM_BUFFER<0,16> EQL MSG$_DELPROC
            THEN ERR_EXIT (.(TERM_BUFFER+4));

            $QIO (
                CHAN = .MAILBOX_CHANNEL,
                FUNC = IO$_READVBLK,
                EFN  = MOUNT_EFN,
                IOSB = IO_STATUS[0],
                P1   = TERM_BUFFER,
                P2   = TERM_BUFFER_LEN
                );
            $SETEF (EFN = MOUNT_EFN);
            END;
        END                                     ! end of ACP wait loop

! If we time out of the loop, the ACP is hung for some reason (such as that
! the image wasn't an ACP). Bomb it and clean up.
!

    THEN
        BEGIN
        $DELPRC (PIDADR = ACP_PID);     ! blow it away using the extended pid
        ERR_EXIT (MOUN$_ACPTIME);
        END;
```

```
! The ACP is really and truly now up. Dispose of the mailbox.
!

    $DASSGN (CHAN = .MAILBOX_CHANNEL);
    CLEANUP_FLAGS[CLF_DEASSMBX] = 0;
    CLEANUP_FLAGS[CLF_DELAQB] = 0;
    END;                                ! end of ACP creation


! Enable the device and issue the MOUNT QIO.
!

UCB[UCB$V_MOUNTING] = 1;
VCB[VCB$L_AQB] = .AQB;

IF NOT .DEVICE_CHAR[DEV$V_SQD]
THEN
    BEGIN
    IF .MOUNT_OPTIONS[OPT_IS_FILES11B]
    THEN SET_DATACHECK (.UCB, HOME_BLOCK)
    ELSE SET_DATACHECK (.UCB, 0);

    !
    ! To prevent the race condition between a simultaneous mount and a
    ! dismount, "deallocate" the device if this is not a private mount.
    ! This ensures that the dismount processing in the file system will
    ! see a consistent I/O database, i.e. UCB$L_PID accurately reflects
    ! how the volume is mounted, even while mount is in progress.
    !
    IF  ( NOT .MOUNT_OPTIONS [OPT_NOSHARE] )            ! and non-private mounts
    AND ( .UCB [UCB$L_PID] NEQ 0 )                       ! and device "allocated", i.e.
    THEN                                                 !    mount is in progress
        BEGIN
        LOCK_IODB ();                                    ! Lock I/O database
        UCB [UCB$L_PID] = 0;                             ! "Deallocate" the device
        BBLOCK [ UCB [UCB$L_DEVCHAR], DEV$V_ALL ] = 0;
        UCB [UCB$W_REFC] = .UCB [UCB$W_REFC] - 1;
        UNLOCK_IODB ();
        END;

    STATUS = DO_IO (CHAN = .CHANNEL,
                    FUNC = IO$_MOUNT,
                    IOSB = IO_STATUS[0]
                    );

    END
ELSE
    BEGIN
    !
    ! Use the 1st device for tapes
    !
    IF (STATUS = $ASSIGN (CHAN = FIRST_CHANNEL, DEVNAM = PHYS_NAME[0]))
    THEN
        STATUS = DO_IO (CHAN = .FIRST_CHANNEL,
                        FUNC = IO$_MOUNT,
                        IOSB = IO_STATUS[0]
                       );
    $DASSGN (CHAN = .FIRST_CHANNEL);
```

```
        END;

    ! If the MOUNT QIO directive fails, just unhook the VCB and drop the mount
    ! count in the AQB. If the AQB count goes to zero, wake the ACP who will
    ! clean himself up in the proper manner. This must be done this way to avoid
    ! timing windows which could conceivably lose I/O packets.
    !

    IF .STATUS THEN STATUS = .IO_STATUS[0];
    IF NOT .STATUS
    THEN
        BEGIN
        LOCK_IODB ();
        UCB[UCB$V_MOUNTING] = 0;
        UCB[UCB$L_VCB] = 0;
        AQB[AQB$B_MNTCNT] = .AQB[AQB$B_MNTCNT] - 1;

        !
        ! We marked the device as "not allocated" prior to issuing
        ! the MOUNT QIO. Now that the QIO failed, restore the UCB
        ! to its original state so that the device deallocation
        ! routine (in module ASSIST) can deallocate this device and
        ! release the device lock.
        !
        IF  ( NOT .MOUNT_OPTIONS [OPT_NOSHARE] )              ! For non-private mounts
        THEN
            BEGIN
            UCB [UCB$L_PID] = .CTL$GL_PCB [PCB$L_PID];       ! Re-allocate the device
            BBLOCK [ UCB [UCB$L_DEVCHAR], DEV$V_ALL ] = 1;
            UCB [UCB$W_REFC] = .UCB [UCB$W_REFC] + 1;
            END;

        UNLOCK_IODB ();
        IF .AQB[AQB$B_MNTCNT] EQL 0 AND NOT .AQB [AQB$V_XQIOPROC]
        THEN

    ! The AQB$L_ACPPID contains an internal pid, we must convert to extended pid to call the
    ! $WAKE service.
    !
            BEGIN
            LINKAGE
                cvt_lnk = JSB (REGISTER=0) : PRESERVE (1,2,3,4,5) NOTUSED (6,7,8,9,10,11);
            EXTERNAL ROUTINE
                EXE$IPID_TO_EPID : cvt_lnk ADDRESSING_MODE (GENERAL);
            ACP_PID = EXE$IPID_TO_EPID (.AQB[AQB$L_ACPPID]);
            $WAKE (PIDADR = ACP_PID);
            END;
        ERR_EXIT (.STATUS);
        END;

    END;                                        ! end of routine START_ACP

                                                        .TITLE  STACP
                                                        .IDENT  \V04-001\

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

```
                        FFFFFFFF  FFB3B4C0  00000 P.AAA:   .LONG    -5000000, -1                         ;
                  45 48 43 41 43 50 51 58  00008 P.AAB:   .ASCII   \XQPCACHE\                           ;
         43 41 21 43 41 21 57 55 21 43 41 21  00010 P.AAD:   .ASCII   \!AC!UW!AC!AC\                       ;
                                 0000000C  0001C P.AAC:   .LONG    12                                   ;
                                 00000000' 00020          .ADDRESS P.AAD                                ;
                                       03  00024 P.AAE:   .BYTE    3                                    ;
                                 50 43 41  00025          .ASCII   \ACP\                                ;
         3A 4D 45 54 53 59 53 24 53 59 53  00028 P.AAF:   .ASCII   \SYS$SYSTEM:\                        ;
                  5D 45 58 45 53 59 53 5B  00033 P.AAG:   .ASCII   \[SYSEXE]\                           ;
                  45 58 45 2E 50 43 41 41  0003B P.AAH:   .ASCII   \AACP.EXE\                           ;
49 4D 52 45 54 5F 47 55 42 45 44 24 50 43 41  00043 P.AAJ:   .ASCII   \ACP$DEBUG_TERMINAL\                 ;
                                 4C 41 4E  00052                                                        ;
                                           00055          .BLKB    3
                                 00000012  00058 P.AAI:   .LONG    18                                   ;
                                 00000000' 0005C          .ADDRESS P.AAJ                                ;
                        FFFFFFFF  FFFFFFFF  00060 P.AAK:   .LONG    -1, -1                               ;
                        FFFFFFFF  FFB3B4C0  00068 P.AAL:   .LONG    -5000000, -1                         ;

                                                          .PSECT   $OWN$,NOEXE,2

                                            00000 OWN_START:
                                                          .BLKB    0
                                            00000 MAILBOX_CHAR:
                                                          .BLKB    16
                                            00010 MAILBOX_DESC:
                                                          .BLKB    8
                                            00018 TERM_BUFFER:
                                                          .BLKB    8
                                            00020 PROC_NAME:
                                                          .BLKB    8
                                            00028 PROCBUF:.BLKB    16
                                            00038 FILE_NAME:
                                                          .BLKB    8
                                            00040 FILEBUF:.BLKB    64
                                            00080 IO_STATUS:
                                                          .BLKB    8
                                            00088 TRANSBUF:
                                                          .BLKB    16
                                            00098 TRANSDESC:
                                                          .BLKB    8
                                            000A0 OWN_END:.BLKB    0

                                                          .EXTRN   PQL$C_SYSPQLLEN
                                                          .EXTRN   CACHE_STATUS, STORED_CONTEXT
                                                          .EXTRN   PHYS_NAME, DEVICE_CHAR
                                                          .EXTRN   MOUNT_OPTIONS, CLEANUP_FLAGS
                                                          .EXTRN   CHANNEL, MAILBOX_CHANNEL
                                                          .EXTRN   HOME_BLOCK, ACP_STRING
                                                          .EXTRN   REAL_RVT, REAL_AQB
                                                          .EXTRN   REAL_WCB, PQL$AB_SYSPQL
                                                          .EXTRN   CTL$GL_PCB, IOC$GL_AQBLIST
                                                          .EXTRN   EXE$GL_FLAGS, ACP$GW_WORKSET
                                                          .EXTRN   ACP$GW_MAPCACHE
                                                          .EXTRN   ACP$GW_HDRCACHE
                                                          .EXTRN   ACP$GW_DIRCACHE
                                                          .EXTRN   ACP$GB_SWAPFLGS
```

```
                                                                .EXTRN   ACP$GB_BASEPRIO
                                                                .EXTRN   EXE$V_MULTACP, EXE$V_INIT
                                                                .EXTRN   ACP$V_SWAPSYS, ACP$V_SWAPGRP
                                                                .EXTRN   ACP$V_SWAPPRV, ACP$V_SWAPMAG
                                                                .EXTRN   LOCK_IODB, UNLOCK_IODB
                                                                .EXTRN   ALLOCATE_MEM, DEALLOCATE_MEM
                                                                .EXTRN   SET_DATACHECK, IOC$CVT_DEVNAM
                                                                .EXTRN   IOC$SEARCHDEV, BUG$_NOTUCBRVT
                                                                .EXTRN   SYS$SETIMR, SYS$WAITFR
                                                                .EXTRN   SYS$CANTIM, SYS$SETEF
                                                                .EXTRN   SYS$FAO, SYS$CREMBX
                                                                .EXTRN   SYS$GETCHN, SYS$TRNLOG
                                                                .EXTRN   SYS$CREPRC, EXE$EPID_TO_IPID
                                                                .EXTRN   SYS$WAKE, SYS$QIO
                                                                .EXTRN   SYS$DELPRC, SYS$DASSGN
                                                                .EXTRN   COMMON_IO, SYS$ASSIGN
                                                                .EXTRN   EXE$IPID_TO_EPID

                                                                .PSECT   $CODE$,NOWRT,2

                                               0FFC 00000       .ENTRY   START_ACP, Save R2,R3,R4,R5,R6,R7,R8,R9,-      ; 0694
                                                                         R10,R11                                        ;
                         5E          88    AE   9E 00002        MOVAB    -120(SP), SP                                   ;
 00A0   8F        00     6E                00   2C 00006        MOVC5    #0, (SP), #0, #160, OWN_START                  ; 0874
                                00000000'  00      0000D                                                                ;
            00000000'    00                10   D0 00012        MOVL     #16, MAILBOX_DESC                              ; 0875
            00000000'    00     00000000'  00   9E 00019        MOVAB    MAILBOX_CHAR, MAILBOX_DESC+4                   ; 0876
            00000000'    00                10   D0 00024        MOVL     #16, TRANSDESC                                 ; 0877
            00000000'    00     00000000'  00   9E 0002B        MOVAB    TRANSBUF, TRANSDESC+4                          ; 0878
                 0000G   CF                01   D0 00036        MOVL     #1, CACHE_STATUS                               ; 0879
                         50          04    AC   D0 0003B        MOVL     UCB, R0                                        ; 0894
                  34     A0          08    AC   D0 0003F        MOVL     VCB, 52(R0)                                    ;
                         59                3C   D0 00044        MOVL     #60, J                                         ; 0897
            00000000G    EF                00   FB 00047 1$:    CALLS    #0, LOCK_IODB                                  ; 0900
                                           58   D4 0004E        CLRL     CREATE_ACP                                     ; 0902
                         51          04    AC   D0 00050        MOVL     UCB, R1                                        ; 0904
                         50          28    A1   D0 00054        MOVL     40(R1), R0                                     ;
                         5A          13    A0   90 00058        MOVB     19(R0), CLASS                                  ;
          55      38     A1                05   E0 0005C        BBS      #5, 56(R1), 7$                                 ; 0910
                                 0000G     CF   D5 00061        TSTL     REAL_RVT                                       ; 0911
                                           4F   13 00065        BEQL     7$                                             ;
                         50      0000G     CF   D0 00067        MOVL     REAL_RVT, R0                                   ;
                         01          04    A0   B1 0006C        CMPW     4(R0), #1                                      ;
                                           44   13 00070        BEQL     7$                                             ;
                         50      0000G     CF   D0 00072        MOVL     REAL_RVT, R0                                   ; 0915
                         53          0B    A0   9A 00077        MOVZBL   11(R0), R3                                     ;
                         52          44    A0   9E 0007B        MOVAB    68(R0), R2                                     ; 0918
                                           50   D4 0007F        CLRL     J                                              ;
                                           0C   11 00081        BRB      3$                                             ;
                         56       FC A240       D0 00083 2$:    MOVL     -4(R2)[J], ACP_UCB                             ;
                                           05   13 00088        BEQL     3$                                             ; 0919
                         51                56   D1 0008A        CMPL     ACP_UCB, R1                                    ;
                                           08   12 0008D        BNEQ     4$                                             ;
          F0             50                53   F3 0008F 3$:    AOBLEQ   R3, J, 2$                                      ; 0915
                                             FEFF  00093        BUGW                                                    ; 0923
                                            0000*  00095        .WORD    <BUG$_NOTUCBRVT!4>                             ;
          04      3A     A6                03   E0 00097 4$:    BBS      #3, 58(ACP_UCB), 5$                            ; 0925
```

```
                                          FEFF 0009C          BUGW                                          ; 0926
                                         0000* 0009E          .WORD   <BUG$_NOTUCBRVT!4>                    ;
                     50        34    A6   D0  000A0  5$:      MOVL    52(ACP_UCB), R0                       ; 0928
                     57        10    A0   D0  000A4           MOVL    16(R0), AQB                           ;
            0C       AC        15    A7   91  000A8           CMPB    21(AQB), TYPE                         ; 0929
                                     04   13  000AD           BEQL    6$                                    ;
                                          FEFF 000AF          BUGW                                          ; 0930
                                         0000* 000B1          .WORD   <BUG$_NOTUCBRVT!4>                    ;
                                     00CD 31  000B3  6$:      BRW     20$                                   ; 0910
       03       0000G CF                  02  E1  000B6  7$:  BBC     #2, MOUNT_OPTIONS+3, 9$               ; 0936
                                     00C1 31  000BC  8$:      BRW     19$                                   ;
       F7       0000G CF                  04  E0  000BF  9$:  BBS     #4, MOUNT_OPTIONS+3, 8$               ;
       76       0000G CF                  03  E1  000C5       BBC     #3, MOUNT_OPTIONS+3, 13$              ; 0943
                     51       0000G  CF   9E  000CB           MOVAB   ACP_STRING, ACP_DEVICE                ; 0948
                     54   00000000G  00   D0  000D0           MOVL    CTL$GL_PCB, R4                        ; 0949
                          00000000G  9F   16  000D7           JSB     @#IOC$SEARCHDEV                       ;
                     6E              50   D0  000DD           MOVL    R0, STATUS                            ;
                     56              51   D0  000E0           MOVL    ACP_DEVICE, ACP_UCB                   ; 0950
                     14              6E   E8  000E3           BLBS    STATUS, 10$                           ; 0952
          00000000G  EF              00   FB  000E6           CALLS   #0, UNLOCK_IODB                       ; 0955
                          00728094   8F   DD  000ED           PUSHL   #7504020                              ; 0956
          00000000G  00              01   FB  000F3           CALLS   #1, LIB$STOP                          ;
       09        3A  A6              03   E1  000FA  10$:     BBC     #3, 58(ACP_UCB), 11$                  ; 0958
       04        39  A6              06   E1  000FF           BBC     #6, 57(ACP_UCB), 11$                  ; 0959
                     14        3B    A6   E9  00104           BLBC    59(ACP_UCB), 12$                      ; 0960
          00000000G  EF              00   FB  00108  11$:     CALLS   #0, UNLOCK_IODB                       ; 0963
                          00728094   8F   DD  0010F           PUSHL   #7504020                              ; 0964
          00000000G  00              01   FB  00115           CALLS   #1, LIB$STOP                          ;
                     50        34    A6   D0  0011C  12$:     MOVL    52(ACP_UCB), R0                       ; 0966
                     57        10    A0   D0  00120           MOVL    16(R0), AQB                           ;
            0C       AC        15    A7   91  00124           CMPB    21(AQB), TYPE                         ; 0967
                                     58   13  00129           BEQL    20$                                   ;
          00000000G  EF              00   FB  0012B           CALLS   #0, UNLOCK_IODB                       ; 0970
                          007280A4   8F   DD  00132           PUSHL   #7504036                              ; 0971
          00000000G  00              01   FB  00138           CALLS   #1, LIB$STOP                          ;
                                     42   11  0013F           BRB     20$                                   ; 0943
                     57   00000000G  9F   D0  00141  13$:     MOVL    @#IOC$GL_AQBLIST, AQB                 ; 0982
 50 00000000G  9F              01       00G EF  00148         EXTZV   S^EXE$V_MULTACP, #1, @#EXE$GL_FLAGS, R0 ; 0985
                                     57   D5  00151  14$:     TSTL    AQB                                   ; 0983
                                     27   13  00153           BEQL    18$                                   ;
                     12              50   E9  00155           BLBC    R0, 15$                               ; 0988
       19            14  A7          01   E1  00158           BBC     #1, 20(AQB), 17$                      ;
                     0C  AC    15    A7   91  0015D           CMPB    21(AQB), TYPE                         ; 0989
                                     12   12  00162           BNEQ    17$                                   ;
                     5A        16    A7   91  00164           CMPB    22(AQB), CLASS                        ; 0990
                                     0A   11  00168           BRB     16$                                   ;
       07            14  A7          02   E1  0016A  15$:     BBC     #2, 20(AQB), 17$                      ; 0996
                     0C  AC    15    A7   91  0016F           CMPB    21(AQB), TYPE                         ; 0997
                                     06   13  00174  16$:     BEQL    18$                                   ;
                     57        10    A7   D0  00176  17$:     MOVL    16(AQB), AQB                          ; 1001
                                     D5   11  0017A           BRB     14$                                   ; 0983
                                     57   D5  0017C  18$:     TSTL    AQB                                   ; 1003
                                     03   12  0017E           BNEQ    20$                                   ;
                     58              01   D0  00180  19$:     MOVL    #1, CREATE_ACP                        ;
                     6F              58   E9  00183  20$:     BLBC    CREATE_ACP, 27$                       ; 1011
                     7E              1C   7D  00186           MOVQ    #28, -(SP)                            ; 1014
                0000G  CF            02   FB  00189           CALLS   #2, ALLOCATE_MEM                      ;
```

```
                    57              50  D0 0018E          MOVL    R0, AQB                                       ;
            0A      A7      0103    8F  B0 00191          MOVW    #259, 10(AQB)                                 ; 1015
            15      A7        0C    AC  90 00197          MOVB    TYPE, 21(AQB)                                 ; 1017
            50      14              A7  9E 0019C          MOVAB   20(AQB), R0                                   ; 1021
   05    0000G      CF              02  E1 001A0          BBC     #2, STORED_CONTEXT, 21$                       ; 1019
                    60              10  88 001A6          BISB2   #16, (R0)                                     ; 1021
                                    03  11 001A9          BRB     22$                                           ;
                    60              08  88 001AB  21$:    BISB2   #8, (R0)                                      ; 1031
   06    0000G      CF              02  E0 001AE  22$:    BBS     #2, MOUNT_OPTIONS+3, 23$                      ; 1033
   05    0000G      CF              04  E1 001B4          BBC     #4, MOUNT_OPTIONS+3, 24$                      ;
                    60              01  88 001BA  23$:    BISB2   #1, (R0)                                      ; 1035
                                    14  11 001BD          BRB     26$                                           ;
   09 00000000G     9F             00G  E1 001BF  24$:    BBC     S^EXE$V_MULTACP, @#EXE$GL_FLAGS, 25$          ; 1036
                    60              02  88 001C7          BISB2   #2, (R0)                                      ; 1039
            16      A7              5A  90 001CA          MOVB    CLASS, 22(AQB)                                ; 1040
                                    03  11 001CE          BRB     26$                                           ; 1036
                    60              04  88 001D0  25$:    BISB2   #4, (R0)                                      ; 1043
                    67              57  D0 001D3  26$:    MOVL    AQB, (AQB)                                    ; 1045
            04      A7              57  D0 001D6          MOVL    AQB, 4(AQB)                                   ; 1046
            10      A7 00000000G    9F  D0 001DA          MOVL    @#IOC$GL_AQBLIST, 16(AQB)                     ; 1047
      00000000G     9F              57  D0 001E2          MOVL    AQB, @#IOC$GL_AQBLIST                         ; 1048
          0000G     CF              57  D0 001E9          MOVL    AQB, REAL_AQB                                 ; 1049
          0000G     CF              04  88 001EE          BISB2   #4, CLEANUP_FLAGS                             ; 1050
                                    5D  11 001F3          BRB     32$                                           ; 1051
   05       14      A7              03  E0 001F5  27$:    BBS     #3, 20(AQB), 28$                              ; 1061
                              0B    A7  96 001FA          INCB    11(AQB)                                       ; 1064
                                    53  11 001FD          BRB     32$                                           ; 1065
      00000000G     EF              00  FB 001FF  28$:    CALLS   #0, UNLOCK_IODB                               ; 1069
                    7E      03E7    8F  3C 00206          MOVZWL  #999, -(SP)                                   ; 1076
                                    7E  D4 0020B          CLRL    -(SP)                                         ;
                            0000'   CF  9F 0020D          PUSHAB  P.AAA                                         ;
                                    1B  DD 00211          PUSHL   #27                                           ;
      00000000G     00              04  FB 00213          CALLS   #4, SYS$SETIMR                                ;
                    20              50  E9 0021A          BLBC    R0, 29$                                       ;
                                    1B  DD 0021D          PUSHL   #27                                           ; 1079
      00000000G     00              01  FB 0021F          CALLS   #1, SYS$WAITFR                                ;
                                    7E  D4 00226          CLRL    -(SP)                                         ; 1080
                    7E      03E7    8F  3C 00228          MOVZWL  #999, -(SP)                                   ;
      00000000G     00              02  FB 0022D          CALLS   #2, SYS$CANTIM                                ;
                                    1B  DD 00234          PUSHL   #27                                           ; 1081
      00000000G     00              01  FB 00236          CALLS   #1, SYS$SETEF                                 ;
                    02              59  F5 0023D  29$:    SOBGTR  J, 30$                                        ; 0897
                                    03  11 00240          BRB     31$                                           ;
                                  FE02  31 00242  30$:    BRW     1$                                            ;
                        00728154    8F  DD 00245  31$:    PUSHL   #7504212                                      ; 1090
      00000000G     00              01  FB 0024B          CALLS   #1, LIB$STOP                                  ;
                    4E              58  E9 00252  32$:    BLBC    CREATE_ACP, 33$                               ; 1094
                    55        04    AC  D0 00255          MOVL    UCB, R5                                       ; 1114
            04      AE        14    A7  9E 00259          MOVAB   20(AQB), 4(SP)                                ; 1097
   4A       04      BE              04  E1 0025E          BBC     #4, @4(SP), 34$                               ;
          0000G     CF              08  88 00263          BISB2   #8, CLEANUP_FLAGS+1                           ; 1104
                            0000G   CF  9F 00268          PUSHAB  CACHE_STATUS                                  ; 1106
          0000V     CF              01  FB 0026C          CALLS   #1, SETUP_BLOCKCACHE                          ;
            18      A7              50  D0 00271          MOVL    R0, 24(AQB)                                   ;
            04      BE              08  8A 00275          BICB2   #8, @4(SP)                                    ; 1108
          0000G     CF      0804    8F  AA 00279          BICW2   #2052, CLEANUP_FLAGS                          ; 1110
                    52        18    A7  D0 00280          MOVL    24(AQB), CACHE_HDR                            ; 1112
```

```
                       51      00AD   C2  9E 00284       MOVAB   173(CACHE_HDR), R1                          ; 1114
                                      54  D4 00289       CLRL    R4                                          ;
                       50             0F  D0 0028B       MOVL    #15, R0                                     ;
                          00000000G   9F  16 0028E       JSB     @#IOC$CVT_DEVNAM                            ;
      00AC    C2       51             08  81 00294       ADDB3   #8, LENGTH, 172(CACHE_HDR)                  ; 1116
      00AD  C142    0000'  CF         08  28 0029A       MOVC3   #8, P.AAB, 173(LENGTH)[CACHE_HDR]           ; 1118
             00000000G  EF            00  FB 002A3  33$: CALLS   #0, UNLOCK_IODB                             ; 1120
                                    0360  31 002AA       BRW     60$                                         ; 1097
             00000000G  EF            00  FB 002AD  34$: CALLS   #0, UNLOCK_IODB                             ; 1136
             00000000'  00            10  D0 002B4       MOVL    #16, PROC_NAME                              ; 1138
             00000000'  00 00000000'  00  9E 002BB       MOVAB   PROCBUF, PROC_NAME+4                        ; 1139
                    08  AE            01  90 002C6       MOVB    #1, ACP_TYPE                                ; 1140
   09     AE        0C  AC     40     8F  81 002CA       ADDB3   #64, TYPE, ACP_TYPE+1                       ; 1141
                               0000'  CF  9F 002D1       PUSHAB  P.AAE                                       ; 1151
                                  0C  AE  9F 002D5       PUSHAB  ACP_TYPE                                    ;
                       7E         54  A5  3C 002D8       MOVZWL  84(R5), -(SP)                               ;
                       58         28  A5  D0 002DC       MOVL    40(R5), R8                                  ;
                                  14  A8  9F 002E0       PUSHAB  20(R8)                                      ;
                           00000000'  00  9F 002E3       PUSHAB  PROC_NAME                                   ;
                           00000000'  00  9F 002E9       PUSHAB  PROC_NAME                                   ;
                               0000'  CF  9F 002EF       PUSHAB  P.AAC                                       ;
             00000000G  00            07  FB 002F3       CALLS   #7, SYS$FAO                                 ;
          0A 00000000G  9F           00G  E1 002FA       BBC     S^EXE$V_INIT, @#EXE$GL_FLAGS, 35$           ; 1153
                       56             0B  D0 00302       MOVL    #11, FILE_PREF_LEN                          ; 1156
                       59      0000'  CF  9E 00305       MOVAB   P.AAF, FILE_PREF                            ; 1157
                                      08  11 0030A       BRB     36$                                         ; 1153
                       56             08  D0 0030C  35$: MOVL    #8, FILE_PREF_LEN                           ; 1161
                       59      0000'  CF  9E 0030F       MOVAB   P.AAG, FILE_PREF                            ; 1162
             00000000'  00 00000000'  00  9E 00314  36$: MOVAB   FILEBUF, FILE_NAME+4                        ; 1165
          32     0000G  CF            04  E1 0031F       BBC     #4, MOUNT_OPTIONS+3, 37$                    ; 1166
             00000000'  00       0000GDF46  9E 00325     MOVAB   @ACP_STRING[FILE_PREF_LEN], FILE_NAME       ; 1169
                       5B 00000000'   00  D0 0032F       MOVL    FILE_NAME, R11                              ; 1171
                       5A 00000000'   00  9E 00336       MOVAB   FILEBUF, R10                                ; 1170
   5B               00                69  56  2C 0033D   MOVC5   FILE_PREF_LEN, (FILE_PREF), #0, R11, (R10)  ;
                                      6A     00342                                                           ;
                                      5B  18 00343       BGEQ    39$                                         ;
                       5A             56  C0 00345       ADDL2   FILE_PREF_LEN, R10                          ;
                       5B             56  C2 00348       SUBL2   FILE_PREF_LEN, R11                          ;
   5B               00       0000G  DF     0000G  CF  2C 0034B   MOVC5   ACP_STRING, @ACP_STRING+4, #0, R11, (R10)   ;
                                      6A     00354                                                           ;
                                      49  11 00355       BRB     39$                                         ; 1166
                       50         0B  A6  9E 00357  37$: MOVAB   11(R6), R0                                  ; 1175
             00000000'  00            50  D0 0035B       MOVL    R0, FILE_NAME                               ;
                       5B             50  D0 00362       MOVL    R0, R11                                     ; 1178
                       5A 00000000'   00  9E 00365       MOVAB   FILEBUF, R10                                ; 1176
   5B               00                69  56  2C 0036C   MOVC5   FILE_PREF_LEN, (FILE_PREF), #0, R11, (R10)  ;
                                      6A     00371                                                           ;
                                      1D  18 00372       BGEQ    38$                                         ;
                       5A             56  C0 00374       ADDL2   FILE_PREF_LEN, R10                          ;
                       5B             56  C2 00377       SUBL2   FILE_PREF_LEN, R11                          ;
   5B               00         10     A8  03  2C 0037A   MOVC5   #3, 16(R8), #0, R11, (R10)                  ;
                                      6A     00380                                                           ;
                                      0E  18 00381       BGEQ    38$                                         ;
                       5A             03  C0 00383       ADDL2   #3, R10                                     ;
                       5B             03  C2 00386       SUBL2   #3, R11                                     ;
   5B               00       0000'    CF  08  2C 00389   MOVC5   #8, P.AAH, #0, R11, (R10)                   ;
                                      6A     00390                                                           ;
```

```
         09      0000G  CF           02  E1 00391 38$:     BBC     #2, MOUNT_OPTIONS+4, 39$                    ; 1179
            00000000'0046        42  8F  90 00397          MOVB    #66, FILEBUF+3[FILE_PREF_LEN]               ; 1180
                           0000G CF  D4 003A0 39$:         CLRL    MAILBOX_CHANNEL                             ; 1187
                                 7E  7C 003A4              CLRQ    -(SP)                                       ; 1191
                      7E    FFED 8F  3C 003A6              MOVZWL  #65517, -(SP)                               ;
                      7E      84 8F  9A 003AB              MOVZBL  #132, -(SP)                                 ;
                      7E      84 8F  9A 003AF              MOVZBL  #132, -(SP)                                 ;
                           0000G CF  9F 003B3              PUSHAB  MAILBOX_CHANNEL                             ;
                                 7E  D4 003B7              CLRL    -(SP)                                       ;
             00000000G  00       07  FB 003B9              CALLS   #7, SYS$CREMBX                              ;
                        6E       50  D0 003C0              MOVL    R0, STATUS                                  ;
                        09       6E  E8 003C3              BLBS    STATUS, 40$                                 ; 1192
                                 6E  DD 003C6              PUSHL   STATUS                                      ;
             00000000G  00       01  FB 003C8              CALLS   #1, LIB$STOP                                ;
                 0000G  CF       08  88 003CF 40$:         BISB2   #8, CLEANUP_FLAGS                           ; 1193
                                 7E  7C 003D4              CLRQ    -(SP)                                       ; 1196
                       00000000' 00  9F 003D6              PUSHAB  MAILBOX_DESC                                ;
                                 7E  D4 003DC              CLRL    -(SP)                                       ;
                           0000G CF  DD 003DE              PUSHL   MAILBOX_CHANNEL                             ;
             00000000G  00       05  FB 003E2              CALLS   #5, SYS$GETCHN                              ;
                        59       04  D0 003E9              MOVL    #4, SWAP_FLAG                               ; 1198
         05      0000G  CF       05  E1 003EC              BBC     #5, DEVICE_CHAR, 41$                        ; 1199
                        50      00G  9A 003F2              MOVZBL  S^ACP$V_SWAPMAG, R0                         ;
                                 22  11 003F5              BRB     45$                                         ;
         0A         04  BE       02  E0 003F7 41$:         BBS     #2, @4(SP), 42$                             ; 1200
         05         04  BE       01  E0 003FC              BBS     #1, @4(SP), 42$                             ;
                        05     0000G CF  E9 00401          BLBC    MOUNT_OPTIONS+1, 43$                        ; 1201
                        50      00G  9A 00406 42$:         MOVZBL  S^ACP$V_SWAPSYS, R0                         ; 1200
                                 0E  11 00409              BRB     45$                                         ;
                           0000G CF  95 0040B 43$:         TSTB    MOUNT_OPTIONS                               ; 1202
                                 05  18 0040F              BGEQ    44$                                         ;
                        50      00G  9A 00411              MOVZBL  S^ACP$V_SWAPGRP, R0                         ;
                                 03  11 00414              BRB     45$                                         ;
                        50      00G  9A 00416 44$:         MOVZBL  S^ACP$V_SWAPPRV, R0                         ;
         02 00000000G   9F       50  E1 00419 45$:         BBC     R0, @#ACP$GB_SWAPFLGS, 46$                  ; 1199
                                 59  D4 00421              CLRL    SWAP_FLAG                                   ; 1205
         06      0000G  CF       05  E1 00423 46$:         BBC     #5, DEVICE_CHAR, 47$                        ; 1210
                        58    80 8F  9A 00429              MOVZBL  #128, WORKING_SET                           ; 1212
                                 27  11 0042D              BRB     48$                                         ;
                58 00000000G     9F  3C 0042F 47$:         MOVZWL  @#ACP$GW_WORKSET, WORKING_SET               ; 1215
                                 1E  12 00436              BNEQ    48$                                         ; 1216
                50 00000000G     9F  3C 00438              MOVZWL  @#ACP$GW_MAPCACHE, R0                       ; 1217
                51 00000000G     9F  3C 0043F              MOVZWL  @#ACP$GW_HDRCACHE, R1                       ;
                        50       51  C0 00446              ADDL2   R1, R0                                      ;
                51 00000000G     9F  3C 00449              MOVZWL  @#ACP$GW_DIRCACHE, R1                       ;
                58      0096 C140    9E 00450              MOVAB   150(R1)[R0], WORKING_SET                    ;
                50 00000000G     8F  D0 00456 48$:         MOVL    #PQL$C_SYSPQLLEN, R0                        ; 1225
             0000005E   8F       50  D1 0045D              CMPL    R0, #94                                     ;
                                 04  15 00464              BLEQ    49$                                         ;
                        50    5E 8F  9A 00466              MOVZBL  #94, R0                                     ;
      14     AE 00000000G  00    50  28 0046A 49$:         MOVC3   R0, PQL$AB_SYSPQL, QUOTA_LIST               ;
                        50    14 AE  9E 00473              MOVAB   QUOTA_LIST, R0                              ; 1226
                50 00000000G     E0  9E 00477              MOVAB   PQL$C_SYSPQLLEN(R0), R0                     ;
                        60       0A  90 0047E              MOVB    #10, (R0)                                   ;
                   01   A0       58  D0 00481              MOVL    WORKING_SET, 1(R0)                          ; 1227
                              05 A0  94 00485              CLRB    5(R0)                                       ; 1228
                                 7E  7C 00488              CLRQ    -(SP)                                       ; 1234
```

```
                                  7E  D4 0048A          CLRL    -(SP)                                ;
                      00000000'   00  9F 0048C          PUSHAB  TRANSDESC                            ;
                      00000000'   00  9F 00492          PUSHAB  TRANSDESC                            ;
                          0000'   CF  9F 00498          PUSHAB  P.AAI                                ;
        00000000G  00             06  FB 0049C          CALLS   #6, SYS$TRNLOG                       ;
                                  7E  D4 004A3          CLRL    -(SP)                                ; 1246
7E                 59             20  C9 004A5          BISL3   #32, SWAP_FLAG, -(SP)                ;
                   7E  00000000'  00  3C 004A9          MOVZWL  MAILBOX_CHAR+12, -(SP)               ;
                       00010003   8F  DD 004B0          PUSHL   #65539                               ;
                   7E  00000000G  9F  9A 004B6          MOVZBL  @#ACP$GB_BASEPRIO, -(SP)             ;
                       00000000'  00  9F 004BD          PUSHAB  PROC_NAME                            ;
                             2C   AE  9F 004C3          PUSHAB  QUOTA_LIST                           ;
                           0000'  CF  9F 004C6          PUSHAB  P.AAK                                ;
                                  7E  7C 004CA          CLRQ    -(SP)                                ;
                       00000000'  00  9F 004CC          PUSHAB  TRANSDESC                            ;
                       00000000'  00  9F 004D2          PUSHAB  FILE_NAME                            ;
                             40   AE  9F 004D8          PUSHAB  ACP_PID                              ;
        00000000G  00             0D  FB 004DB          CALLS   #13, SYS$CREPRC                      ;
                   6E             50  D0 004E2          MOVL    R0, STATUS                           ;
                   09             6E  E8 004E5          BLBS    STATUS, 50$                          ; 1248
                                  6E  DD 004E8          PUSHL   STATUS                               ;
        00000000G  00             01  FB 004EA          CALLS   #1, LIB$STOP                         ;
                   50        10   AE  D0 004F1 50$:     MOVL    ACP_PID, R0                          ; 1259
                       00000000G  00  16 004F5          JSB     EXE$EPID_TO_IPID                     ;
              0C   A7             50  D0 004FB          MOVL    R0, 12(A0B)                          ;
                                  7E  D4 004FF          CLRL    -(SP)                                ; 1261
                             14   AE  9F 00501          PUSHAB  ACP_PID                              ;
        00000000G  00             02  FB 00504          CALLS   #2, SYS$WAKE                         ;
                   6E             50  D0 0050B          MOVL    R0, STATUS                           ;
                   09             6E  E8 0050E          BLBS    STATUS, 51$                          ; 1262
                                  6E  DD 00511          PUSHL   STATUS                               ;
        00000000G  00             01  FB 00513          CALLS   #1, LIB$STOP                         ;
                                  7E  7C 0051A 51$:     CLRQ    -(SP)                                ; 1276
                                  7E  7C 0051C          CLRQ    -(SP)                                ;
                                  08  DD 0051E          PUSHL   #8                                   ;
                       00000000'  00  9F 00520          PUSHAB  TERM_BUFFER                          ;
                                  7E  7C 00526          CLRQ    -(SP)                                ;
                       00000000'  00  9F 00528          PUSHAB  IO_STATUS                            ;
                                  31  DD 0052E          PUSHL   #49                                  ;
                           0000G  CF  DD 00530          PUSHL   MAILBOX_CHANNEL                      ;
                                  1A  DD 00534          PUSHL   #26                                  ;
        00000000G  00             0C  FB 00536          CALLS   #12, SYS$QIO                         ;
                                  1A  DD 0053D          PUSHL   #26                                  ; 1277
        00000000G  00             01  FB 0053F          CALLS   #1, SYS$SETEF                        ;
                   52             3C  D0 00546          MOVL    #60, J                               ; 1279
                   7E        03E7 8F  3C 00549 52$:     MOVZWL  #999, -(SP)                          ; 1281
                                  7E  D4 0054E          CLRL    -(SP)                                ;
                           0000'  CF  9F 00550          PUSHAB  P.AAL                                ;
                                  1B  DD 00554          PUSHL   #27                                  ;
        00000000G  00             04  FB 00556          CALLS   #4, SYS$SETIMR                       ;
                   20             50  E9 0055D          BLBC    R0, 53$                              ;
                                  1B  DD 00560          PUSHL   #27                                  ; 1284
        00000000G  00             01  FB 00562          CALLS   #1, SYS$WAITFR                       ;
                                  7E  D4 00569          CLRL    -(SP)                                ; 1285
                   7E        03E7 8F  3C 0056B          MOVZWL  #999, -(SP)                          ;
        00000000G  00             02  FB 00570          CALLS   #2, SYS$CANTIM                       ;
                                  1B  DD 00577          PUSHL   #27                                  ; 1286
```

```
      00000000G  00              01  FB 00579        CALLS   #1, SYS$SETEF                      ;
 78         04   BE              03  E1 00580 53$:   BBC     #3, @4(SP), 59$                    ; 1289
                 50  00000000'   00  D0 00585        MOVL    IO_STATUS, R0                      ; 1292
                                 4E  13 0058C        BEQL    56$                                ;
                 09              50  E8 0058E        BLBS    R0, 54$                            ; 1295
                                 50  DD 00591        PUSHL   R0                                 ;
      00000000G  00              01  FB 00593        CALLS   #1, LIB$STOP                       ;
                 03  00000000'   00  B1 0059A 54$:   CMPW    TERM_BUFFER, #3                    ; 1296
                                 0D  12 005A1        BNEQ    55$                                ;
                     00000000'   00  DD 005A3        PUSHL   TERM_BUFFER+4                      ; 1297
      00000000G  00              01  FB 005A9        CALLS   #1, LIB$STOP                       ;
                                 7E  7C 005B0 55$:   CLRQ    -(SP)                              ; 1306
                                 7E  7C 005B2        CLRQ    -(SP)                              ;
                                 08  DD 005B4        PUSHL   #8                                 ;
                     00000000'   00  9F 005B6        PUSHAB  TERM_BUFFER                        ;
                                 7E  7C 005BC        CLRQ    -(SP)                              ;
                     00000000'   00  9F 005BE        PUSHAB  IO_STATUS                          ;
                                 31  DD 005C4        PUSHL   #49                                ;
                          0000G  CF  DD 005C6        PUSHL   MAILBOX_CHANNEL                    ;
                                 1A  DD 005CA        PUSHL   #26                                ;
      00000000G  00              0C  FB 005CC        CALLS   #12, SYS$QIO                       ;
                                 1A  DD 005D3        PUSHL   #26                                ; 1307
      00000000G  00              01  FB 005D5        CALLS   #1, SYS$SETEF                      ;
                 02              52  F5 005DC 56$:   SOBGTR  J, 57$                             ; 1279
                                 03  11 005DF        BRB     58$                                ;
                           FF65  31  005E1 57$:      BRW     52$                                ;
                                 7E  D4 005E4 58$:   CLRL    -(SP)                              ; 1317
                           14    AE  9F 005E6        PUSHAB  ACP_PID                            ;
      00000000G  00              02  FB 005E9        CALLS   #2, SYS$DELPRC                     ;
                       0072814C  8F  DD 005F0        PUSHL   #7504204                           ; 1318
      00000000G  00              01  FB 005F6        CALLS   #1, LIB$STOP                       ;
                          0000G  CF  DD 005FD 59$:   PUSHL   MAILBOX_CHANNEL                    ; 1324
      00000000G  00              01  FB 00601        CALLS   #1, SYS$DASSGN                     ;
          0000G  CF              0C  8A 00608        BICB2   #12, CLEANUP_FLAGS                 ; 1326
                 52        04    AC  D0 0060D 60$:   MOVL    UCB, R2                            ; 1333
            65   A2              02  88 00611        BISB2   #2, 101(R2)                        ;
                 50        08    AC  D0 00615        MOVL    VCB, R0                            ; 1334
            10   A0              57  D0 00619        MOVL    AQB, 16(R0)                        ;
 5B       0000G  CF              05  E0 0061D        BBS     #5, DEVICE_CHAR, 64$               ; 1336
 06       0000G  CF              02  E1 00623        BBC     #2, MOUNT_OPTIONS+4, 61$           ; 1339
                          0000G  CF  9F 00629        PUSHAB  HOME_BLOCK                         ; 1340
                                 02  11 0062D        BRB     62$                                ;
                                 7E  D4 0062F 61$:   CLRL    -(SP)                              ; 1341
                                 52  DD 00631 62$:   PUSHL   R2                                 ;
          0000G  CF              02  FB 00633        CALLS   #2, SET_DATACHECK                  ;
 1E       0000G  CF              04  E0 00638        BBS     #4, MOUNT_OPTIONS, 63$             ; 1350
                           2C    A2  D5 0063E        TSTL    44(R2)                             ; 1351
                                 19  13 00641        BEQL    63$                                ;
      00000000G  EF              00  FB 00643        CALLS   #0, LOCK_IODB                      ; 1354
                           2C    A2  D4 0064A        CLRL    44(R2)                             ; 1355
            3A   A2        80    8F  8A 0064D        BICB2   #128, 58(R2)                       ; 1356
                           5C    A2  B7 00652        DECW    92(R2)                             ; 1357
      00000000G  EF              00  FB 00655        CALLS   #0, UNLOCK_IODB                    ; 1358
                                 7E  7C 0065C 63$:   CLRQ    -(SP)                              ; 1364
                                 7E  7C 0065E        CLRQ    -(SP)                              ;
                                 7E  7C 00660        CLRQ    -(SP)                              ;
                                 7E  7C 00662        CLRQ    -(SP)                              ;
```

```
                          00000000'  00  9F 00664         PUSHAB  IO_STATUS                          ;
                                     39  DD 0066A         PUSHL   #57                                ;
                              0000G  CF  DD 0066C         PUSHL   CHANNEL                            ;
                                     1A  DD 00670         PUSHL   #26                                ;
              00000000G  00          0C  FB 00672         CALLS   #12, COMMON_IO                     ;
                         6E          50  D0 00679         MOVL    R0, STATUS                         ;
                                     3F  11 0067C         BRB     66$                                ;
                                     7E  7C 0067E 64$:    CLRQ    -(SP)                              ; 1336
                                 14  AE  9F 00680         PUSHAB  FIRST_CHANNEL                      ; 1371
                              0000G  CF  9F 00683         PUSHAB  PHYS_NAME                          ;
              00000000G  00          04  FB 00687         CALLS   #4, SYS$ASSIGN                     ;
                         6E          50  D0 0068E         MOVL    R0, STATUS                         ;
                         1F          6E  E9 00691         BLBC    STATUS, 65$                        ;
                                     7E  7C 00694         CLRQ    -(SP)                              ; 1376
                                     7E  7C 00696         CLRQ    -(SP)                              ;
                                     7E  7C 00698         CLRQ    -(SP)                              ;
                                     7E  7C 0069A         CLRQ    -(SP)                              ;
                          00000000'  00  9F 0069C         PUSHAB  IO_STATUS                          ;
                                     39  DD 006A2         PUSHL   #57                                ;
                                 34  AE  DD 006A4         PUSHL   FIRST_CHANNEL                      ;
                                     1A  DD 006A7         PUSHL   #26                                ;
              00000000G  00          0C  FB 006A9         CALLS   #12, COMMON_IO                     ;
                         6E          50  D0 006B0         MOVL    R0, STATUS                         ;
                                 0C  AE  DD 006B3 65$:    PUSHL   FIRST_CHANNEL                      ; 1377
              00000000G  00          01  FB 006B6         CALLS   #1, SYS$DASSGN                     ;
                         0A          6E  E9 006BD 66$:    BLBC    STATUS, 67$                        ; 1386
                         6E 00000000' 00 D0 006C0         MOVL    IO_STATUS, STATUS                  ;
                         5F          6E  E8 006C7         BLBS    STATUS, 70$                        ; 1387
              00000000G  EF          00  FB 006CA 67$:    CALLS   #0, LOCK_IODB                      ; 1390
                     65  A2          02  8A 006D1         BICB2   #2, 101(R2)                        ; 1391
                                 34  A2  D4 006D5         CLRL    52(R2)                             ; 1392
                                 0B  A7  97 006D8         DECB    11(AQB)                            ; 1393
        14        0000G  CF          04  E0 006DB         BBS     #4, MOUNT_OPTIONS, 68$             ; 1402
                         50 00000000G 00 D0 006E1         MOVL    CTL$GL_PCB, R0                     ; 1405
                     2C  A2      60  A0  D0 006E8         MOVL    96(R0), 44(R2)                     ;
                     3A  A2      80  8F  88 006ED         BISB2   #128, 58(R2)                       ; 1406
                                 5C  A2  B6 006F2         INCW    92(R2)                             ; 1407
              00000000G  EF          00  FB 006F5 68$:    CALLS   #0, UNLOCK_IODB                    ; 1410
                                 0B  A7  95 006FC         TSTB    11(AQB)                            ; 1411
                                     1F  12 006FF         BNEQ    69$                                ;
        1A           14  A7          04  E0 00701         BBS     #4, 20(AQB), 69$                   ;
                         50      0C  A7  D0 00706         MOVL    12(AQB), R0                        ; 1422
                          00000000G  00  16 0070A         JSB     EXE$IPID_TO_EPID                   ;
                     10  AE          50  D0 00710         MOVL    R0, ACP_PID                        ;
                                     7E  D4 00714         CLRL    -(SP)                              ; 1423
                                 14  AE  9F 00716         PUSHAB  ACP_PID                            ;
              00000000G  00          02  FB 00719         CALLS   #2, SYS$WAKE                       ;
                                     6E  DD 00720 69$:    PUSHL   STATUS                             ; 1425
              00000000G  00          01  FB 00722         CALLS   #1, LIB$STOP                       ;
                                         04 00729 70$:    RET                                        ; 1428

; Routine Size:  1834 bytes,    Routine Base:  $CODE$ + 0000

;  899          1429  1
```

```
1 ROUTINE SETUP_BLOCKCACHE (STATUS) =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine allocates dynamic memory and initializes it for
1 !       use as the block cache for the AQB desired.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       SETUP_BLOCKCACHE (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !
1 !       ARG1 :  Address of a longword to receive the cache
1 !               allocation status
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       ARG1 :  1 if the block cache allocated
1 !               0 if a reduced block cache is allocated
1 !
1 ! ROUTINE VALUE:
1 !
1 !       Address of initialized buffer cache.
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2 BEGIN
2
2 MACRO
2       QA (SIZE) = (((SIZE) + 15) AND NOT 15) %,       ! Quad Align
2       FLNK = 0,0,32,0 %,
2       BLNK = 4,0,32,0 %;
2
2 BUILTIN INSQUE;
2
2 EXTERNAL ROUTINE
2       ALLOCATE_MEM;
2
2 EXTERNAL
2       ACP$GW_MAPCACHE : WORD ADDRESSING_MODE (GENERAL),
2       ACP$GW_HDRCACHE : WORD ADDRESSING_MODE (GENERAL),
2       ACP$GW_DIRCACHE : WORD ADDRESSING_MODE (GENERAL),
2       ACP$GW_DINDXCACHE : WORD ADDRESSING_MODE (GENERAL);
2
2 LOCAL
2       MAP_COUNT,
2       HDR_COUNT,
2       DIR_COUNT,
```

```
        DINDX_COUNT,
        BFRCOUNT,
        BUFFER_BYTES,
        OVRHD_BYTES,
        TOTAL_BYTES,
        EOMEM,
        HSHTBLS,
        INDX,
        BFRD            : REF BLOCKVECTOR [,BFRD$S_BFRDDEF,BYTE],
        BFRL            : REF BLOCKVECTOR [,BFRL$S_BFRLDEF,BYTE],
        CACHE_HDR       : REF BBLOCK;

.STATUS = 1;                                    ! Assume succesful allocation
MAP_COUNT = MAXU (1, .ACP$GW_MAPCACHE);
HDR_COUNT = MAXU (3, .ACP$GW_HDRCACHE);
DIR_COUNT = MAXU (2, .ACP$GW_DIRCACHE);
DINDX_COUNT = MAXU (1, .ACP$GW_DINDXCACHE);

BFRCOUNT = .MAP_COUNT + .HDR_COUNT + .DIR_COUNT + .DINDX_COUNT;

IF .BFRCOUNT GTRU (2^15 - 2)
THEN
    BEGIN
    MAP_COUNT = (2^15/6) - 1;
    DINDX_COUNT = (2^15/6) - 1;
    HDR_COUNT = (2^16/6) - 1;
    DIR_COUNT = (2^16/6) - 1;
    BFRCOUNT = .MAP_COUNT + .HDR_COUNT + .DIR_COUNT + .DINDX_COUNT;
    END;

BUFFER_BYTES = .BFRCOUNT*512;

OVRHD_BYTES = QA (F11BC$S_F11BCDEF)             ! Overall cache header
            + QA (.BFRCOUNT*BFRD$S_BFRDDEF)     ! plus descriptors
            + QA (.BFRCOUNT*BFRL$S_BFRLDEF)     ! plus buffer lock blocks
            + QA (.BFRCOUNT*2)                  ! plus LBN hash table
            + QA (.BFRCOUNT*2);                 ! plus buffer lock hash table.

! Add up total requirements, plus add 1 page so we can line the buffers
! up on page boundaries.
!

TOTAL_BYTES = .BUFFER_BYTES + .OVRHD_BYTES + 512;

CACHE_HDR = ALLOCATE_PAGED (.TOTAL_BYTES);      ! Allocate cache from paged pool

IF .CACHE_HDR EQL 0
THEN
    BEGIN
    !
    ! The full block cache allocation has failed due to insufficient paged
    ! pool. Retry to allocate a reduced block cache with the following:
    !
    !   MAP_COUNT = 2;
    !   HDR_COUNT = 6;
    !   DIR_COUNT = 4;
    !   DINDX_COUNT = 2;
```

```
	!
	! If the reduced block cache is successfully allocated, set the proper
	! status in the argument of this routine. Otherwise, the allocation
	! routine will signal an error, and the mount will fail with an "insuf-
	! ficient dynamic memory" error.
	!
	MAP_COUNT = 2;
	HDR_COUNT = 6;
	DIR_COUNT = 4;
	DINDX_COUNT = 2;
	BFRCOUNT = .MAP_COUNT + .HDR_COUNT + .DIR_COUNT + .DINDX_COUNT;
	BUFFER_BYTES = .BFRCOUNT*512;
	OVRHD_BYTES = QA (F11BC$S_F11BCDEF)		! Overall cache header
		    + QA (.BFRCOUNT*BFRD$S_BFRDDEF)	! plus descriptors
		    + QA (.BFRCOUNT*BFRL$S_BFRLDEF)	! plus buffer lock blocks
		    + QA (.BFRCOUNT*2)			! plus LBN hash table
		    + QA (.BFRCOUNT*2);			! plus buffer lock hash table.
	TOTAL_BYTES = .BUFFER_BYTES + .OVRHD_BYTES + 512;
	CACHE_HDR = ALLOCATE_MEM (.TOTAL_BYTES,1);	! Allocate cache from paged pool
	.STATUS = 0;					! Set reduced cache status
	END;

CACHE_HDR [F11BC$L_REALSIZE] = .(CACHE_HDR [F11BC$W_SIZE])<0,32>;

%IF F11BC$K_NUM_POOLS NEQ 4
%THEN %WARN ('Bad pool count constant');
%FI

BEGIN

BIND
	POOL_LRU	= CACHE_HDR [F11BC$Q_POOL_LRU] : BLOCKVECTOR [,8,BYTE],
	POOL_WAITQ	= CACHE_HDR [F11BC$Q_POOL_WAITQ] : BLOCKVECTOR [,8,BYTE],
	POOLCNT		= CACHE_HDR [F11BC$W_POOLCNT] : VECTOR [,WORD],
	POOLAVAIL	= CACHE_HDR [F11BC$L_POOLAVAIL] : VECTOR;

POOLCNT [0] = POOLAVAIL [0] = .MAP_COUNT;
POOLCNT [1] = POOLAVAIL [1] = .DIR_COUNT;
POOLCNT [2] = POOLAVAIL [2] = .HDR_COUNT;
POOLCNT [3] = POOLAVAIL [3] = .DINDX_COUNT;

CACHE_HDR [F11BC$W_BFRCNT] = .BFRCOUNT;

CACHE_HDR [F11BC$W_SIZE] = F11BC$S_F11BCDEF;
CACHE_HDR [F11BC$B_TYPE] = DYN$C_PGD;
CACHE_HDR [F11BC$B_SUBTYPE] = DYN$C_PGD_F11BC;

EOMEM = (.CACHE_HDR + .CACHE_HDR [F11BC$L_REALSIZE]) AND NOT 511;

CACHE_HDR [F11BC$L_BUFBASE] = .EOMEM - .BUFFER_BYTES;
CACHE_HDR [F11BC$L_BUFSIZE] = .BUFFER_BYTES;

BFRD = (CACHE_HDR [F11BC$L_BFRDBAS] = QA (.CACHE_HDR + F11BC$S_F11BCDEF));

BFRL = (CACHE_HDR [F11BC$L_BFRLDBAS] = QA (.BFRD + .BFRCOUNT*BFRD$S_BFRDDEF));

HSHTBLS = .BFRL + .BFRCOUNT*BFRL$S_BFRLDEF;
```

```
3 CACHE_HDR [F11BC$L_LBNHSHBAS] = .HSHTBLS;
3 CACHE_HDR [F11BC$W_LBNHSHCNT] = (.CACHE_HDR [F11BC$L_BUFBASE] - .HSHTBLS)/4;
3
3 CACHE_HDR [F11BC$L_BLHSHBAS] = .CACHE_HDR [F11BC$L_LBNHSHBAS]
3                                + .CACHE_HDR [F11BC$W_LBNHSHCNT]*2;
3 CACHE_HDR [F11BC$W_BLHSHCNT] = .CACHE_HDR [F11BC$W_LBNHSHCNT];
3
3 ! Run through all the buffer descriptors and link them into the appropriate
3 ! pool, noting in each descriptor which pool it belongs to.
3 !
3
3 CACHE_HDR [F11BC$W_FREEBFRL] = 1;
3
3 INDX = 0;
3
3 INCR POOL FROM 0 TO 3
3 DO
4     BEGIN
4
4     POOL_LRU [.POOL, FLNK] = POOL_LRU [.POOL, FLNK];
4     POOL_LRU [.POOL, BLNK] = POOL_LRU [.POOL, FLNK];
4
4     POOL_WAITQ [.POOL, FLNK] = POOL_WAITQ [.POOL, FLNK];
4     POOL_WAITQ [.POOL, BLNK] = POOL_WAITQ [.POOL, FLNK];
4
4     INCR I FROM 0 TO .POOLCNT [.POOL] - 1
4     DO
5         BEGIN
5         INSQUE (BFRD [.INDX, BFRD$L_QFL], .POOL_LRU [.POOL, BLNK]);
5         BFRD [.INDX, BFRD$V_POOL] = .POOL;
5         BFRL [.INDX, BFRL$W_NXTBFRL] = .INDX + 2;
5         INDX = .INDX + 1;
4         END;
3     END;
3
3 BFRL [.INDX-1, BFRL$W_NXTBFRL] = 0;
3
3 .CACHE_HDR
3
3 END
1 END;


                                                        .EXTRN  ACP$GW_DINDXCACHE

                                07FC 00000 SETUP_BLOCKCACHE:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 1430
        04    BC                01 D0 00002             MOVL    #1, @STATUS                             ; 1499
              50 00000000G      00 3C 00006             MOVZWL  ACP$GW_MAPCACHE, R0                     ; 1500
                                03 12 0000D             BNEQ    1$                                      ;
              50                01 D0 0000F             MOVL    #1, R0                                  ;
              55                50 D0 00012 1$:         MOVL    R0, MAP_COUNT                           ;
              50 00000000G      00 3C 00015             MOVZWL  ACP$GW_HDRCACHE, R0                     ; 1501
              03                50 B1 0001C             CMPW    R0, #3                                  ;
                                03 1E 0001F             BGEQU   2$                                      ;
              50                03 D0 00021             MOVL    #3, R0                                  ;
```

```
                         5A        50  D0 00024 2$:   MOVL    R0, HDR_COUNT                            ;
                   50 00000000G    00  3C 00027       MOVZWL  ACP$GW_DIRCACHE, R0                      ; 1502
                         02        50  B1 0002E       CMPW    R0, #2                                   ;
                                   03  1E 00031       BGEQU   3$                                       ;
                         50        02  D0 00033       MOVL    #2, R0                                   ;
                         59        50  D0 00036 3$:   MOVL    R0, DIR_COUNT                            ;
                   50 00000000G    00  3C 00039       MOVZWL  ACP$GW_DINDXCACHE, R0                    ; 1503
                                   03  12 00040       BNEQ    4$                                       ;
                         50        01  D0 00042       MOVL    #1, R0                                   ;
                         58        50  D0 00045 4$:   MOVL    R0, DINDX_COUNT                          ;
         50              55        5A  C1 00048       ADDL3   HDR_COUNT, MAP_COUNT, R0                 ; 1505
                         50        59  C0 0004C       ADDL2   DIR_COUNT, R0                            ;
         52              50        58  C1 0004F       ADDL3   DINDX_COUNT, R0, BFRCOUNT                ;
             0000FFFE    8F        52  D1 00053       CMPL    BFRCOUNT, #65534                         ; 1507
                                   1F  1B 0005A       BLEQU   5$                                       ;
                         55   2AA9 8F  3C 0005C       MOVZWL  #10921, MAP_COUNT                        ; 1510
                         58   2AA9 8F  3C 00061       MOVZWL  #10921, DINDX_COUNT                      ; 1511
                         5A   5554 8F  3C 00066       MOVZWL  #21844, HDR_COUNT                        ; 1512
                         59   5554 8F  3C 0006B       MOVZWL  #21844, DIR_COUNT                        ; 1513
         50              55        5A  C1 00070       ADDL3   HDR_COUNT, MAP_COUNT, R0                 ; 1514
                         50        59  C0 00074       ADDL2   DIR_COUNT, R0                            ;
         52              50        58  C1 00077       ADDL3   DINDX_COUNT, R0, BFRCOUNT                ;
         56              52        09  78 0007B 5$:   ASHL    #9, BFRCOUNT, BUFFER_BYTES               ; 1517
         51              52        05  78 0007F       ASHL    #5, BFRCOUNT, R1                         ; 1520
                         53     0F A1  9E 00083       MOVAB   15(R1), R3                               ;
                         53        0F  8A 00087       BICB2   #15, R3                                  ;
         51              52        04  78 0008A       ASHL    #4, BFRCOUNT, R1                         ; 1521
                         51        0F  C0 0008E       ADDL2   #15, R1                                  ;
                         51        0F  8A 00091       BICB2   #15, R1                                  ;
         50              53        51  C1 00094       ADDL3   R1, R3, R0                               ;
         53              52        01  78 00098       ASHL    #1, BFRCOUNT, R3                         ; 1522
                         53        0F  C0 0009C       ADDL2   #15, R3                                  ;
                         53        0F  8A 0009F       BICB2   #15, R3                                  ;
                         50        53  C0 000A2       ADDL2   R3, R0                                   ;
                         54 00D0 C340  9E 000A5       MOVAB   208(R3)[R0], OVRHD_BYTES                 ; 1523
                         57 0200 C446  9E 000AB       MOVAB   512(OVRHD_BYTES)[BUFFER_BYTES], TOTAL_BYTES ; 1529
                                   57  DD 000B1       PUSHL   TOTAL_BYTES                              ; 1531
                 0000V   CF        01  FB 000B3       CALLS   #1, ALLOCATE_PAGED                       ;
                                   50  D5 000B8       TSTL    CACHE_HDR                                ; 1533
                                   58  12 000BA       BNEQ    6$                                       ;
                         55        02  D0 000BC       MOVL    #2, MAP_COUNT                            ; 1550
                         5A        06  D0 000BF       MOVL    #6, HDR_COUNT                            ; 1551
                         59        04  D0 000C2       MOVL    #4, DIR_COUNT                            ; 1552
                         58        02  D0 000C5       MOVL    #2, DINDX_COUNT                          ; 1553
         51              55        5A  C1 000C8       ADDL3   HDR_COUNT, MAP_COUNT, R1                 ; 1554
                         51        59  C0 000CC       ADDL2   DIR_COUNT, R1                            ;
         52              51        58  C1 000CF       ADDL3   DINDX_COUNT, R1, BFRCOUNT                ;
         56              52        09  78 000D3       ASHL    #9, BFRCOUNT, BUFFER_BYTES               ; 1555
         51              52        05  78 000D7       ASHL    #5, BFRCOUNT, R1                         ; 1557
                         53     0F A1  9E 000DB       MOVAB   15(R1), R3                               ;
                         53        0F  8A 000DF       BICB2   #15, R3                                  ;
         51              52        04  78 000E2       ASHL    #4, BFRCOUNT, R1                         ; 1558
                         51        0F  C0 000E6       ADDL2   #15, R1                                  ;
                         51        0F  8A 000E9       BICB2   #15, R1                                  ;
                         51        53  C0 000EC       ADDL2   R3, R1                                   ;
         53              52        01  78 000EF       ASHL    #1, BFRCOUNT, R3                         ; 1559
                         53        0F  C0 000F3       ADDL2   #15, R3                                  ;
```

```
                    53              0F  8A 000F6          BICB2   #15, R3                                          ;
                    51              53  C0 000F9          ADDL2   R3, R1                                           ;
                    54    00D0 C341     9E 000FC          MOVAB   208(R3)[R1], OVRHD_BYTES                         ; 1560
                    57    0200 C446     9E 00102          MOVAB   512(OVRHD_BYTES)[BUFFER_BYTES], TOTAL_BYTES      ; 1561
                                    01  DD 00108          PUSHL   #1                                               ; 1562
                                    57  DD 0010A          PUSHL   TOTAL_BYTES                                      ;
          0000G  CF                 02  FB 0010C          CALLS   #2, ALLOCATE_MEM                                 ;
                              04    BC  D4 00111          CLRL    @STATUS                                          ; 1563
             0C  A0           08    A0  D0 00114  6$:     MOVL    8(CACHE_HDR), 12(CACHE_HDR)                      ; 1566
                 54           78    A0  9E 00119          MOVAB   120(CACHE_HDR), R4                               ; 1577
                 51           68    A0  9E 0011D          MOVAB   104(CACHE_HDR), R1                               ; 1578
                 61                 55  D0 00121          MOVL    MAP_COUNT, (R1)                                  ; 1580
                 64                 55  B0 00124          MOVW    MAP_COUNT, (R4)                                  ;
             04  A1                 59  7D 00127          MOVQ    DIR_COUNT, 4(R1)                                 ; 1581
             02  A4                 59  B0 0012B          MOVW    DIR_COUNT, 2(R4)                                 ;
             04  A4                 5A  B0 0012F          MOVW    HDR_COUNT, 4(R4)                                 ; 1582
             0C  A1                 58  D0 00133          MOVL    DINDX_COUNT, 12(R1)                              ; 1583
             06  A4                 58  B0 00137          MOVW    DINDX_COUNT, 6(R4)                               ;
             16  A0                 52  B0 0013B          MOVW    BFRCOUNT, 22(CACHE_HDR)                          ; 1585
             08  A0 016600C4        8F  D0 0013F          MOVL    #23462084, 8(CACHE_HDR)                          ; 1587
        51       50           0C    A0  C1 00147          ADDL3   12(CACHE_HDR), CACHE_HDR, R1                     ; 1591
                 51         01FF    8F  AA 0014C          BICW2   #511, EOMEM                                      ;
        60       51                 56  C3 00151          SUBL3   BUFFER_BYTES, EOMEM, (CACHE_HDR)                 ; 1593
             04  A0                 56  D0 00155          MOVL    BUFFER_BYTES, 4(CACHE_HDR)                       ; 1594
                 51         00D3    C0  9E 00159          MOVAB   211(R0), R1                                      ; 1596
                 51                 0F  8A 0015E          BICB2   #15, R1                                          ;
             18  A0                 51  D0 00161          MOVL    R1, 24(CACHE_HDR)                                ;
                 56                 51  D0 00165          MOVL    R1, BFRD                                         ;
        53       52                 05  78 00168          ASHL    #5, BFRCOUNT, R3                                 ; 1598
                 51         0F A346     9E 0016C          MOVAB   15(R3)[BFRD], R1                                 ;
                 51                 0F  8A 00171          BICB2   #15, R1                                          ;
             1C  A0                 51  D0 00174          MOVL    R1, 28(CACHE_HDR)                                ;
                 53                 51  D0 00178          MOVL    R1, BFRL                                         ;
                 52                 10  C4 0017B          MULL2   #16, R2                                          ; 1600
        51       52                 53  C1 0017E          ADDL3   BFRL, R2, HSHTBLS                                ;
             10  A0                 51  D0 00182          MOVL    HSHTBLS, 16(CACHE_HDR)                           ; 1601
        51       60                 51  C3 00186          SUBL3   HSHTBLS, (CACHE_HDR), R1                         ; 1602
        52       51                 04  C7 0018A          DIVL3   #4, R1, R2                                       ;
             14  A0                 52  B0 0018E          MOVW    R2, 20(CACHE_HDR)                                ;
                 51           14    A0  3C 00192          MOVZWL  20(CACHE_HDR), R1                                ; 1605
             20  A0         10 B041     3E 00196          MOVAW   @16(CACHE_HDR)[R1], 32(CACHE_HDR)                ; 1606
             24  A0           14    A0  B0 0019C          MOVW    20(CACHE_HDR), 36(CACHE_HDR)                     ;
             26  A0                 01  B0 001A1          MOVW    #1, 38(CACHE_HDR)                                ; 1612
                                    51  7C 001A5          CLRQ    INDX                                             ; 1614
                 55         28 A042     7E 001A7  7$:     MOVAQ   40(CACHE_HDR)[POOL], R5                          ; 1620
                 65                 55  D0 001AC          MOVL    R5, (R5)                                         ;
                            2C A042     7F 001AF          PUSHAQ  44(CACHE_HDR)[POOL]                              ; 1621
                 9E                 55  D0 001B3          MOVL    R5, @(SP)+                                       ;
                 55         48 A042     7E 001B6          MOVAQ   72(CACHE_HDR)[POOL], R5                          ; 1623
                 65                 55  D0 001BB          MOVL    R5, (R5)                                         ;
                            4C A042     7F 001BE          PUSHAQ  76(CACHE_HDR)[POOL]                              ; 1624
                 9E                 55  D0 001C2          MOVL    R5, @(SP)+                                       ;
                 58             6442    3C 001C5          MOVZWL  (R4)[POOL], R8                                   ; 1626
                 57                 01  CE 001C9          MNEGL   #1, I                                            ;
                                    26  11 001CC          BRB     9$                                               ;
        55       51                 05  78 001CE  8$:     ASHL    #5, INDX, R5                                     ; 1629
                 59         2C A042     7E 001D2          MOVAQ   44(CACHE_HDR)[POOL], R9                          ;
```

```
                  00   B9        6546  0E 001D7         INSQUE  (R5)[BFRD], @0(R9)                 ;
            55         51          05  78 001DC         ASHL    #5, INDX, R5                       ; 1630
    18 A546 02         00          52  F0 001E0         INSV    POOL, #0, #2, 24(R5)[BFRD]         ;
            55         51          04  78 001E7         ASHL    #4, INDX, R5                       ; 1631
                                 6543  9F 001EB         PUSHAB  (R5)[BFRL]                         ;
            9E         51          02  A1 001EE         ADDW3   #2, INDX, @(SP)+                   ;
                                   51  D6 001F2         INCL    INDX                               ; 1632
            D6         57          58  F2 001F4  9$:    AOBLSS  R8, I, 8$                          ; 1626
            AB         52          03  F3 001F8         AOBLEQ  #3, POOL, 7$                       ; 1616
                       51          10  C4 001FC         MULL2   #16, R1                            ; 1636
                              F0 A143  9F 001FF         PUSHAB  -16(R1)[BFRL]                      ;
                                   9E  B4 00203         CLRW    @(SP)+                             ;
                                       04 00205         RET                                        ; 1641

; Routine Size:  518 bytes,    Routine Base:  $CODE$ + 072A

; 1113          1642  1
```

```
1
1 ROUTINE ALLOCATE_PAGED (SIZE_NEEDED) =
1
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This is a local paged pool allocation routine for module STACP.
1 !       Functionally, it is identical to the ALLOCATE_MEM routine. The
1 !       difference is that on an allocation failure, this routine return
1 !       a 0 to the caller rather than signalling an error.
1 !
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ALLOCATE_PAGED (ARG1)
1 !
1 ! INPUT PARAMETERS:
1 !
1 !       ARG1: size needed in bytes
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None.
1 !
1 ! OUTPUT PARAMETERS:
1 !
1 !       None.
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       Address of block allocated. If the allocation failed, the routine
1 !       value returned is 0.
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Memory allocated, zeroed, and full longword size is stored
1 !       at size word location.
1 !
1 !--
1
2 BEGIN
2
2 LINKAGE
2       EXE_ALLOCO      = JSB :
2                         NOPRESERVE (3, 4, 5)
2                         GLOBAL (SIZE = 1, ADDRESS = 2);
2
2 LOCAL
2       STATUS,                         ! status return of allocator
2       BLOCK_SIZE,                     ! local copy of size allocated
2       BLOCK_ADDRESS;                  ! local copy of address
2
```

```
; 1172      1700  3 BEGIN                                      ! nested block to avoid scope conflicts
; 1173      1701  3
; 1174      1702  3 GLOBAL REGISTER
; 1175      1703  3         SIZE            = 1,               ! rounded up allocation size
; 1176      1704  3         ADDRESS         = 2 : REF VECTOR [,WORD]; ! address returned by exec routines
; 1177      1705  3
; 1178      1706  3 EXTERNAL ROUTINE
; 1179      1707  3         EXE$ALOPAGED    : EXE_ALLOCO ADDRESSING_MODE (ABSOLUTE);
; 1180      1708  3
; 1181      1709  3
; 1182      1710  3 ! Simply compute the size needed rounded up to the next quadword and call the
; 1183      1711  3 ! appropriate exec allocation routine.
; 1184      1712  3 !
; 1185      1713  3
; 1186      1714  3 SIZE = .SIZE_NEEDED;
; 1187      1715  3
; 1188      1716  3 STATUS = EXE$ALOPAGED ();
; 1189      1717  3
; 1190      1718  3 ! Copy the block size and address into locals to dodge the MOVC5.
; 1191      1719  3 !
; 1192      1720  3
; 1193      1721  3 BLOCK_SIZE = .SIZE;
; 1194      1722  3 BLOCK_ADDRESS = .ADDRESS;
; 1195      1723  2 END;
; 1196      1724  2
; 1197      1725  2 IF NOT .STATUS
; 1198      1726  2 THEN
; 1199      1727  2     RETURN 0;
; 1200      1728  2
; 1201      1729  2 CH$FILL (0, .BLOCK_SIZE, .BLOCK_ADDRESS);
; 1202      1730  2 (.BLOCK_ADDRESS + 8) = .BLOCK_SIZE;
; 1203      1731  2 RETURN .BLOCK_ADDRESS;
; 1204      1732  2
; 1205      1733  1 END;                                       ! end of routine ALLOCATE_MEM


                                                      .EXTRN  EXE$ALOPAGED

                                  0FFC 00000 ALLOCATE_PAGED:
                                                      .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11           ; 1644
                    51       04   AC  D0 00002        MOVL    SIZE_NEEDED, SIZE                              ; 1714
                      00000000G   9F  16 00006        JSB     @#EXE$ALOPAGED                                 ; 1716
                    57            51  D0 0000C        MOVL    SIZE, BLOCK_SIZE                               ; 1721
                    56            52  D0 0000F        MOVL    ADDRESS, BLOCK_ADDRESS                         ; 1722
                    0E            50  E9 00012        BLBC    STATUS, 1$                                     ; 1725
    57        00    6E            00  2C 00015        MOVC5   #0, (SP), #0, BLOCK_SIZE, (BLOCK_ADDRESS)      ; 1729
                    66                   0001A                                                               ;
              08    A6            57  D0 0001B        MOVL    BLOCK_SIZE, 8(BLOCK_ADDRESS)                   ; 1730
                    50            56  D0 0001F        MOVL    BLOCK_ADDRESS, R0                              ; 1731
                                  04     00022        RET
                              50  D4     00023 1$:    CLRL    R0                                             ; 1733
                                  04     00025        RET                                                    ;

; Routine Size:  38 bytes,    Routine Base:  $CODE$ + 0930
```

```
; 1206          1734  1
; 1207          1735  1 END
; 1208          1736  0 ELUDOM



                                                        .EXTRN  LIB$STOP

;                             PSECT SUMMARY

;       Name                      Bytes                       Attributes

;   $OWN$                              160  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $PLIT$                             112  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                            2390  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       100         0         1000           00:02.0



;                             COMMAND QUALIFIERS

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

; Size:           2390 code + 272 data bytes
; Run Time:          00:50.5
; Elapsed Time:      01:57.6
; Lines/CPU Min:      2064
; Lexemes/CPU-Min: 23705
; Memory Used:  476 pages
; Compilation Complete
```

0246 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RDHOME
LIS

MWTUR2
LIS

RUJMAN
LIS

MWTUR1
LIS

SRCVOL
LIS

STACP
LIS

MRDBLK
LIS

REBUILD
LIS

0247 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

VMOUNT
LIS

MPCLRPFM
LIS

MPAST
LIS

MP

MP
MAP

MP
MDL

TRNLOG
LIS

MPCMOD
LIS

MPMACROS
MAR