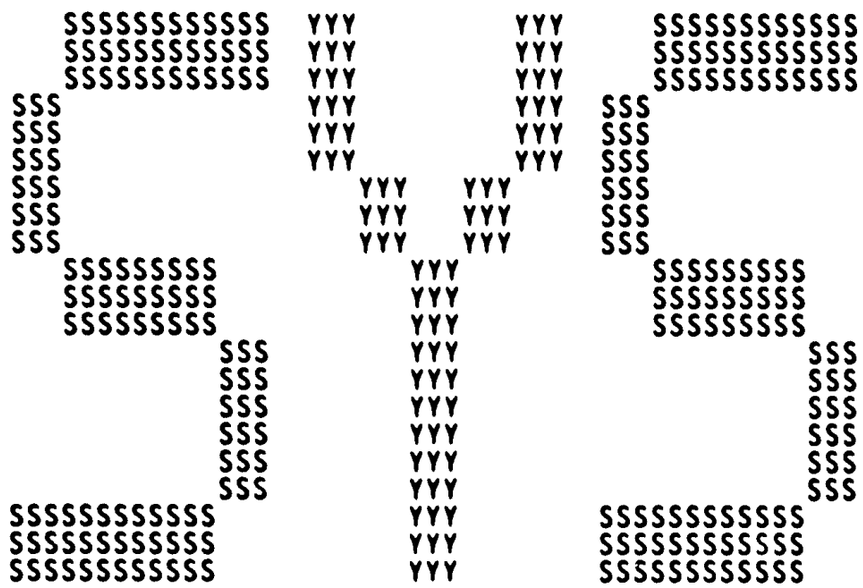

```
IIIIII    NN      NN    IIIIII    TTTTTTTTTT
IIIIII    NN      NN    IIIIII    TTTTTTTTTT
  II      NN      NN      II          TT
  II      NN      NN      II          TT
  II      NNNN    NN      II          TT
  II      NNNN    NN      II          TT
  II      NN  NN  NN      II          TT
  II      NN  NN  NN      II          TT
  II      NN    NNNN      II          TT
  II      NN    NNNN      II          TT
  II      NN      NN      II          TT          ....
  II      NN      NN      II          TT          ....
IIIIII    NN      NN    IIIIII        TT          ....
IIIIII    NN      NN    IIIIII        TT          ....

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

Table of contents

```
0000     1          .TITLE  INIT PROCESSOR INITIALIZATION
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
0000     9 ;*                                                                           *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    11 ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE           *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER      *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14 ;*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY        *
0000    15 ;*  TRANSFERRED.                                                             *
0000    16 ;*                                                                           *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    18 ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
0000    19 ;*  CORPORATION.                                                             *
0000    20 ;*                                                                           *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23 ;*                                                                           *
0000    24 ;*                                                                           *
0000    25 ;*****************************************************************************
0000    26 ;
0000    27 ; SYSTEM INITIALIZATION
0000    28 ;
0000    29 ; D. HUSTVEDT 17-SEP-76
0000    30 ;
0000    31 ; MODIFIED BY:
0000    32 ;
0000    33 ;       V03-054 WMC0054         Wayne Cardoza           06-Aug-1984
0000    34 ;               Tie IRPMIN to SRPSIZE.
0000    35 ;
0000    36 ;       V03-053 WMC0053         Wayne Cardoza           30-Jul-1984
0000    37 ;               New minimum size for IRP list.
0000    38 ;
0000    39 ;       V03-052 TCM0017         Trudy C. Matthews       24-Jul-1984
0000    40 ;               Use the RPB$B_CTRLLTR field when determining the boot
0000    41 ;               device's controller letter.
0000    42 ;
0000    43 ;       V03-051 MSH0068         Michael S. Harvey       23-Jul-1984
0000    44 ;               Correctly initialize the HWTYPE field of the local SB.
0000    45 ;
0000    46 ;       V03-050 WHM0001         Bill Matthews           09-Jul-1984
0000    47 ;               Load SYSLOAxxx before during any I/O to the console terminal.
0000    48 ;               The console terminal I/O routines are now in SYSLOAxxx.
0000    49 ;
0000    50 ;       V03-049 LMP0275         L. Mark Pilant,         12-Jul-1984  19:57
0000    51 ;               Initialize the ACL info in the ORB to be a null descriptor
0000    52 ;               list rather than an empty queue.  This avoids the overhead
0000    53 ;               of locking and unlocking the ACL mutex, only to find out
0000    54 ;               that the ACL was empty.
0000    55 ;
0000    56 ;       V03-048 WMC0048         Wayne Cardoza           05-Jul-1984
0000    57 ;               Pay attention to the SRPMIN parameter.
```

```
0000     58 ;               Eliminate the check of LRPMIN against 480 - this used to be so
0000     59 ;               that PQBs fit.
0000     60 ;               Add consistency checking of size against MIN for SRP, LRP.
0000     61 ;
0000     62 ;       V03-047 ROW0367         Ralph O. Weber          21-MAY-1984
0000     63 ;               Cause IDB$W_UNITS to be 1 in IDB created for system disk
0000     64 ;               class driver.
0000     65 ;
0000     66 ;       V03-046 ROW0363         Ralph O. Weber          11-MAY-1984
0000     67 ;               Force DEV$M_NNM to be set before IOC$CVT_DEVNAM is called to
0000     68 ;               build the translation for SYS$SYSDEVICE.  This is required to
0000     69 ;               cause IOC$CVT_DEVNAM to prepend the node name or allocation
0000     70 ;               class name field.
0000     71 ;
0000     72 ;       V03-045 ROW0358         Ralph O. Weber          3-MAY-1984
0000     73 ;               Return to using -1 in the call to IOC$CVT_DEVNAM when building
0000     74 ;               SYS$SYSDEVICE.  The allocation class is not correctly known
0000     75 ;               during INIT.  An allocation class device name translation for
0000     76 ;               SYS$SYSDEVICE will have to be formed later in the booting
0000     77 ;               process.
0000     78 ;
0000     79 ;       V03-044 ROW0352         Ralph O. Weber          27-APR-1984
0000     80 ;               Change name conversion flag argument from -1 (a displayable
0000     81 ;               device name) to zero (an allocation class device name) in call
0000     82 ;               to IOC$CVT_DEVNAM used to build SYS$SYSDEVICE logical name
0000     83 ;               translation.  This causes the translation for SYS$SYSDEVICE to
0000     84 ;               be useable over time and various cluster failures.  For class
0000     85 ;               drivers, cause a real IDB to be allocated an initialized. This
0000     86 ;               corrects a misformed I/O database problem which causes the
0000     87 ;               SYSGEN command SHOW /DEVICE to crash systems and generally
0000     88 ;               eliminates maintenance headaches in this area.  Also remove
0000     89 ;               reference to the obsolete $LOGDEF.
0000     90 ;
0000     91 ;       V03-043 LMP0240         L. Mark Pilant,         26-Apr-1984  8:45
0000     92 ;               Make sure an ORB is created for the UCB when booting from an
0000     93 ;               HSC device.
0000     94 ;
0000     95 ;       V03-042 EMD0079         Ellen M. Dusseault      11-Apr-1984
0000     96 ;               Store base address of SYSLOA image in MMG$GL_SYSLOA_BASE.
0000     97 ;
0000     98 ;       V03-041 KTA3113         Kerbey T. Altmann       15-Mar-1984
0000     99 ;               Fill in UCB$W_MSCPUNIT for HSC/UDA/Emulated disks.
0000    100 ;
0000    101 ;       V03-040 KTA3109         Kerbey T. Altmann       11-Mar-1984
0000    102 ;               Fill in more fields in permanent local SB.
0000    103 ;
0000    104 ;       V03-039 MMD0251         Meg Dumont,     27-Feb-1984  17:50
0000    105 ;               Add support for $MTACCESS installation specific accessibility
0000    106 ;               routine
0000    107 ;
0000    108 ;       V03-038 ROW0311         Ralph O. Weber          26-FEB-1984
0000    109 ;               Make the page setup for use by mount verification a different
0000    110 ;               page from the black hole page.  Because mount verification
0000    111 ;               will soon be performing read-modify-write operations on the
0000    112 ;               storage control block, it must have a private page of working
0000    113 ;               storage.
0000    114 ;
```

```
0000    115 ;       V03-037 ROW0274         Ralph O. Weber          5-JAN-1984
0000    116 ;               Fix setup of UCB address in IDB$L_UCBLST for system disk IDB.
0000    117 ;               The system disk IDB is built assuming a unit number no greater
0000    118 ;               than 7.  This is not true for MSCP supported system disks.
0000    119 ;               Since the MSCP disk class driver does not used IDB$L_UCBLST,
0000    120 ;               simply do not setup the field when the unit number exceeds 7.
0000    121 ;
0000    122 ;       V03-036 SRB0105         Steve Beckhardt         11-Nov-1983
0000    123 ;               Added initialization of LCK$GB_HTBLSHFT.
0000    124 ;
0000    125 ;       V03-035 ACG0372         Andrew C. Goldstein,    11-Nov-1983  10:32
0000    126 ;               Change page protection of paged pool to ERKW
0000    127 ;
0000    128 ;       V03-034 TCM0016         Trudy C. Matthews       22-Sep-1983
0000    129 ;               Store CLU$GL_ALLOCLS in system disk's DDB.
0000    130 ;
0000    131 ;       V03-033 ROW0207         Ralph O. Weber          13-AUG-1983
0000    132 ;               Establish page protection of UR on all PTEs in the erase
0000    133 ;               psuedo page-table.  This corrects a bug which was causing the
0000    134 ;               CI port to crash because the PTEs contained a page protection
0000    135 ;               of KW.
0000    136 ;
0000    137 ;       V03-032 WMC0032         Wayne Cardoza           06-Aug-1983
0000    138 ;               Remove setting of cluster-wide bit for system disk.
0000    139 ;               Fix register destroyed by misplaced MOVC5.
0000    140 ;
0000    141 ;       V03-031 RAS0175         Ron Schaefer            28-Jul-1983
0000    142 ;               Change definitions of SYS$SYSDEVICE and SYS$DISK
0000    143 ;               to be TERMINAL/CONCEALED rather than use "_"s.
0000    144 ;
0000    145 ;       V03-030 KDM0062         Kathleen D. Morse       18-Jul-1983
0000    146 ;               Move initialization of time-wait data cells to a
0000    147 ;               loadable, cpu-dependent routine, EXE$INI_TIMWAIT.
0000    148 ;
0000    149 ;       V03-029 TCM0015         Trudy C. Matthews       28-Jul-1983
0000    150 ;               Change "BBS" to "BBSS" in KTA3068.
0000    151 ;
0000    152 ;       V03-028 KTA3068         Kerbey T. Altmann       06-Jul-1983
0000    153 ;               Stuff EXE$GQ_TODCBASE into localsb as incarnation number.
0000    154 ;               Remove CPU-specific code in console/boot device init.
0000    155 ;               Add support for booting off MSCP emulated disks.
0000    156 ;               Make system disk always potentially cluster-wide.
0000    157 ;
0000    158 ;       V03-027 KTA3060         Kerbey T. Altmann       22-Jun-1983
0000    159 ;               Add support for boot device name passed in from
0000    160 ;               SYSBOOT.  Also make HSC disk always on 'A' controller.
0000    161 ;
0000    162 ;       V03-026 DMW4053         DMWalp                  21-Jun-1983
0000    163 ;               Convert boot device logical name from CRELOG to CRELNM
0000    164 ;
0000    165 ;       V03-025 TCM0014         Trudy C. Matthews       10-Jun-1983
0000    166 ;               Fix comment in TCM0013.
0000    167 ;
0000    168 ;       V03-024 TCM0013         Trudy C. Matthews       21-Apr-1983
0000    169 ;               Add new input register (R4) to call to IOC$CVT_DEVNAM.
0000    170 ;
0000    171 ;       V03-024 KDM0044         Kathleen D. Morse       03-May-1983
```

```
0000   172 ;                Hook instruction emulation code into SCB, if required.
0000   173 ;
0000   174 ;        V03-023 DWT0093         David W. Thiel          11-Apr-1983
0000   175 ;                Initialize local system block before initializing
0000   176 ;                CLSLOA.
0000   177 ;
0000   178 ;        V03-022 CWH1002         CW Hobbs                24-Feb-1983
0000   179 ;                Compute SCH$GL_PIXWIDTH cell from max process count, init
0000   180 ;                extended pid cells of null and swapper processes.
0000   181 ;
0000   182 ;        V03-021 TCM0012         Trudy C. Matthews       15-Feb-1983
0000   183 ;                Update CPUDISP cases to include 790-specific path.
0000   184 ;
0000   185 ;        V03-020 DWT0072         David W. Thiel          28-Jan-1983
0000   186 ;                Initialize loadable cluster code after SCS loadable code.
0000   187 ;
0000   188 ;        V03-019 STJ3057         Steven T. Jeffreys      21-Jan-1983
0000   189 ;                - Added code to link $ERAPAT loadable code to vectors in system.
0000   190 ;                - Added code to create system Erase Pattern Buffer (EPB) and
0000   191 ;                  the Psuedo Page Table (PPT) to map it.
0000   192 ;
0000   193 ;        V03-018 SRB0059         Steve Beckhardt         6-Jan-1983
0000   194 ;                Added code to link cluster loadable code to vectors
0000   195 ;                in system.
0000   196 ;
0000   197 ;        V03-017 KTA3022         Kerbey T. Altmann       29-Dec-1982
0000   198 ;                Initialize perm local system block.  Add setting
0000   199 ;                of scs nodename into boot device name, if applicable.
0000   200 ;
0000   201 ;        V03-015 SRB0057         Steve Beckhardt         16-Dec-1982
0000   202 ;                Changed initialization of lockid table to store
0000   203 ;                a sequence number of 1 in the 2nd word of each entry.
0000   204 ;
0000   205 ;        V03-014 TCM0011         Trudy C. Matthews       16-Dec-1982
0000   206 ;                Initialize R2 before calling CON$SENDCONSCMD.
0000   207 ;
0000   208 ;        V03-013 DMW4018         DMWalp                  15-Dec-1982
0000   209 ;                Combined CRELOG logical name blocks, and pass it via
0000   210 ;                LNM pointer
0000   211 ;
0000   212 ;        V03-012 TCM0010         Trudy C. Matthews       10-Nov-1982
0000   213 ;                Use new input values in call to CON$SENDCONSCMD.
0000   214 ;
0000   215 ;        V03-011 DMW4005         DMWalp                  10-Nov-1982
0000   216 ;                Recode logical name blocks for SYS$DISK and SYS$SYSDEVICE
0000   217 ;                to use external interface ( not internal ) of $CRELOG
0000   218 ;
0000   219 ;        V03-010 KTA3018         Kerbey T. Altmann       05-Nov-1982
0000   220 ;                Delete loading of INILOA.
0000   221 ;
0000   222 ;        V03-009 BLS0190         Benn Schreiber          19-Oct-1982
0000   223 ;                Ensure console terminal is not autobaud and scope
0000   224 ;
0000   225 ;        V03-008 TCM0009         Trudy C. Matthews       12-Oct-1982
0000   226 ;                Added a delay loop (to avoid saturating the Unibus) to the
0000   227 ;                calibration of the TIMEWAIT macro loop.
0000   228 ;
```

```
0000   229 ;       V03-007 WMC0001         Wayne Cardoza           28-Sep-1982
0000   230 ;               Give RPB a PFN data base.
0000   231 ;
0000   232 ;       V03-006 STJ3022         Steven Jeffreys         22-Sep-1982
0000   233 ;               Renamed routine LINK_VEC to EXE$LINK_VEC and moved it
0000   234 ;               to its own module, LINKVEC.  Note that EXE$LINK_VEC
0000   235 ;               is in a psect that is removed from the system's
0000   236 ;               address space by INIT, and is therefore unavailable
0000   237 ;               for use, in the EXEC, after INIT finishes.
0000   238 ;
0000   239 ;       V03-005 BLS0183         Benn Schreiber          16-Aug-1982
0000   240 ;               Changes for loadable console support
0000   241 ;
0000   242 ;       V03-004 SRB0051         Steve Beckhardt         1-Jun-1982
0000   243 ;               Fixed bug that occurred if the DEADLOCK_WAIT system
0000   244 ;               parameter was zero at boot time and then later set non-zero.
0000   245 ;
0000   246 ;
```

```
0000    248 ; SYSTEM INITIALIZATION
0000    249 ;
0000    250 ; MACRO LIBRARY CALLS
0000    251 ;
0000    252
0000    253         $ADPDEF                         ; DEFINE ADP OFFSETS
0000    254         $ARCDEF                         ; DEFINE ARCHITECTURE BITS
0000    255         $BOODEF                         ; DEFINE BOOT CONTROL BLOCK OFFSETS
0000    256         $BTDDEF                         ; DEFINE BOOT DEVICE TYPES
0000    257         $CONDEF                         ; DEFINE CONSOLE FUNCTION CODES
0000    258         $CRBDEF                         ; DEFINE CRB OFFSETS
0000    259         $DCDEF                          ; DEFINE DEVICE CHARACTERISTICS
0000    260         $DDBDEF                         ; DEFINE DDB OFFSETS
0000    261         $DDTDEF                         ; DEFINE DDT OFFSETS
0000    262         $DEVDEF                         ; DEFINE DEVICE BITS
0000    263         $DPTDEF                         ; DEFINE DRIVER PROLOGUE
0000    264         $DYNDEF                         ; DEFINE DATA STRUCTURE TYPE CODES
0000    265         $IDBDEF                         ; DEFINE IDB OFFSETS
0000    266         $IPLDEF                         ; IPL DEFINITIONS
0000    267         $IRPDEF                         ; DEFINE IRP OFFSETS
0000    268         $LNMDEF                         ; DEFINE LNM OFFSETS
0000    269         $MPMDEF                         ; DEFINE MULTI-PORT MEMORY
0000    270         $MSCPDEF                        ; DEFINE MSCP OFFSETS
0000    271         $NDTDEF                         ; DEFINE NEXUS-DEVICE TYPE CODES
0000    272         $ORBDEF                         ; OBJECT'S RIGHTS BLOCK OFFSETS
0000    273         $PCBDEF                         ; DEFINE PCB OFFSETS
0000    274         $PFLDEF                         ; PAGE FILE OFFSET DEFINITIONS
0000    275         $PFNDEF                         ; PFN DATA BASE OFFSET DEFINITIONS
0000    276         $PHDDEF                         ; DEFINE PROCESS HEADER OFFSETS
0000    277         $PRDEF                          ; DEFINE IPR NUMBERS
0000    278         $PRTDEF                         ; DEFINE PAGE PROTECTION CODES
0000    279         $PTEDEF                         ; PAGE TABLE ENTRY DEFINITIONS
0000    280         $PTRDEF                         ; POINTER CONTROL BLOCK OFFSETS
0000    281         $RBMDEF                         ; Define realtime SPT bit map
0000    282         $RPBDEF                         ; DEFINE RESTART PARAMETER BLOCK
0000    283         $SBDEF                          ; Define system block offsets
0000    284         $SECDEF                         ; SECTION TABLE OFFSET DEFINITIONS
0000    285         $UBADEF                         ; DEFINE UBA REGISTER OFFSETS
0000    286         $UBIDEF                         ; DEFINE UNIBUS INTERCONNECT
0000    287                                         ;  REGISTER OFFSETS
0000    288         $UCBDEF                         ; DEFINE UCB OFFSETS
0000    289         $TTYDEFS                        ; DEFINE TERMINAL DRIVER OFFSETS
0000    290         $TTDEF                          ; TERMINAL ATTRIBUTES
0000    291         $TT2DEF                         ; MORE TERMINAL-SPECIFIC ATTRIBUTES
0000    292         $VADEF                          ; DEFINE VIRTUAL ADDRESS FIELDS
0000    293         $VECDEF                         ; DEFINE VEC OFFSETS
0000    294         $WCBDEF                         ; DEFINE WINDOW CONTROL BLOCK OFFSETS
0000    295 ;
0000    296 ; LOCAL MACROS
0000    297 ;
0000    298 ;
0000    299 ; DEFINE RANGE OF PURE CODE/DATA MACRO:
0000    300 ;
0000    301         .MACRO  PURE,START,STOP         ;
0000    302         .IF     NB      START
0000    303         .LONG   <START-<1@31>>@-7       ;
0000    304         .IFF
```

```
0000    305          .LONG   0
0000    306          .ENDC
0000    307          .IF     NB      STOP
0000    308          .LONG   <STOP-^X80000200>@-7    ;
0000    309          .IFF
0000    310          .LONG   0
0000    311          .ENDC
0000    312          .ENDM   PURE                    ;
```

```
                     0000   314 ;
                     0000   315 ; LOCAL SYMBOLS
                     0000   316 ;
                     0000   317 ; CHARACTER DEFINITIONS
                     0000   318 ;
                     0000   319
0000000D             0000   320 CR=13                                   ; CARRIAGE RETURN
0000000A             0000   321 LF=10                                   ; LINE FEED
0000000C             0000   322 DYN$C_HEADLEN=12                        ; LENGTH OF A SELF-IDENT HEADER
                     0000   323 ;
                     0000   324 ; LOCAL DATA
                     0000   325 ;
                     0000   326
                00000000    327          .PSECT  Z$DEBUGA,PAGE         ; PSECT TO MARK BASE OF XDELTA
                     0000   328 XDELTABASE:                             ;
                00000000    329          .PSECT  Z$INITOCO,LONG        ; Psect to mark base of bootstrap
                     0000   330 EXE$A_BOOPARAM::                       ; parameter block.
                00000000    331          .PSECT  Z$INIT,PAGE
                     0000   332 INI_BASE:                              ;
```

```
                                0000   334 ;
                                0000   335 ; The following Logical Name Structures must be contiguous
                                0000   336 ;
                                0000   337 ; Boot Device Logical name equivalence
                                0000   338 ;
                                0000   339 BDL$GL_DISK_LOG::                          ; item list for SYS$DISK $CRELNM
                                0000   340                                          ;     in SWAPPER init
                         0003 0004  0000   341          .WORD   4,LNM$_ATTRIBUTES     ; size of attributes and item code
                          00000004  0004   342 BDL_L_DISK_AT_PTR = . - BDL$GL_DISK_LOG ; attribute pointer offset
                          00000030' 0004   343          .LONG   110$  - BDL$GL_DISK_LOG ; pointer to SYS$DISK attributes
                          00000000  0008   344          .LONG   0
                          0000000C  000C   345 BDL_W_DISK_EQ_SZ  = . - BDL$GL_DISK_LOG ; equiv name size offset
                         0002 0000  000C   346          .WORD   0,LNM$_STRING         ; size of string and item code
                          00000010  0010   347 BDL_L_DISK_EQ_PTR = . - BDL$GL_DISK_LOG ; equiv name string pointer offset
                          0000001D' 0010   348          .LONG   100$  - BDL$GL_DISK_LOG ; pointer to SYS$DISK equiv name string
                  00000000 00000000  0014   349          .QUAD   0                     ; end of item list
                               5F    001C   350          .BYTE   ^A\_\                 ; space for the leading "_"
                          0000001D  001D   351 BDL_L_DISK_EQV  =   .  - BDL$GL_DISK_LOG ; equiv name string offset
30 30 43 55 44 24 4E 4E 4E 4E 4E 4E  001D   352 100$:   .ASCII  /NNNNNN$DUC000:/       ; equiv name string nnnnnn$ducu[uu]:
                            3A 30    0029
                          00000000  002B   353          .LONG   0                     ; safty area
                          00000030  002F   354          .BLKB   <<<.-BDL$GL_DISK_LOG>+15>&^C<15>>-<.-BDL$GL_DISK_LOG> ; quad
                          00000200  0030   355 110$:    .LONG   LNM$M_TERMINAL        ; terminal attr
                                0034   356
                          00000034  0034   357 BDL$L_SYSDLOG  == .   - BDL$GL_DISK_LOG ; item list for SYS$SYSDEVICE $CRELNM
                                0034   358                                          ;     in SWAPPER init
                         0003 0004  0034   359          .WORD   4,LNM$_ATTRIBUTES     ; size of attributes and item code
                          00000038  0038   360 BDL_L_SYSD_AT_PTR = . - BDL$GL_DISK_LOG ; attribute pointer offset
                          00000070' 0038   361          .LONG   210$  - BDL$GL_DISK_LOG ; pointer to SYS$SYSDEVICE attributes
                          00000000  003C   362          .LONG   0
                          00000040  0040   363 BDL_W_SYSD_EQ_SZ =   . - BDL$GL_DISK_LOG ; equiv name  size offset
                         0002 0000  0040   364          .WORD   0,LNM$_STRING         ; size of string and :tem code
                          00000044  0044   365 BDL_L_SYSD_EQ_PTR = . - BDL$GL_DISK_LOG ; equiv name string pointer offset
                          00000051' 0044   366          .LONG   200$  - BDL$GL_DISK_LOG ; ptr to SYS$SYSDEVICE equiv name str
                  00000000 00000000  0048   367          .QUAD   0                     ; end of item list
                               5F    0050   368          .BYTE   ^A\_\                 ; space for the leading "_"
                          00000051  0051   369 BDL_L_SYSD_EQV  =   .  - BDL$GL_DISK_LOG ; equiv name string offset
30 30 43 55 44 24 4E 4E 4E 4E 4E 4E  0051   370 200$:   .ASCII  /NNNNNN$DUC000:/       ; equiv name string nnnnnn$ducu[uu]:
                            3A 30    005D
                          00000000  005F   371          .LONG   0                     ; safty area
                          00000070  0063   372          .BLKB   <<<.-BDL$GL_DISK_LOG>+15>&^C<15>>-<.-BDL$GL_DISK_LOG>; quad
                          00000200  0070   373 210$:    .LONG   LNM$M_TERMINAL        ; terminal attr
                                0074   374
                          00000074  0074   375 BDL$S_CRELNM_ITMLST == . - BDL$GL_DISK_LOG
```

```
                0074   377 ;
                0074   378 ; MEMORY MANAGEMENT DATA                        ;
                0074   379 ;
                0074   380 PGDCOD:
                0074   381 ;       .LONG   <MMG$AL_PGDCOD-<1@31>>@-9 ; 1ST VPN OF PAGED CODE
00000000        0074   382         .LONG   0                           ; CELL IS LOADED BY INIT
FFC00000'       0078   383         .LONG   <MMG$AL_PGDCODEN-<1@31>>@-9 ; LAST + 1 VPN OF PAGED CODE
7C400000        007C   384         .LONG   <PTE$C_UR ! PTE$C_KOWN ! PTE$M_TYP1 ! PTE$M_TYP0>
                0080   385                                             ; SECTION 0 PAGE TABLE ENTRY
00000074        0080   386         PGDCODBEG=PGDCOD                    ; BEGINNING OF PAGED CODE
00000078        0080   387         PGDCODEND=PGDCOD+4                  ; LAST + 1 PAGE OF PAGED CODE
                0080   388                                             ; SECTION 0 PAGE TABLE ENTRY
                0080   389 PAGEDYN:
00000000        0080   390         .LONG   0                           ; 1ST VPN OF PAGED DYNAMIC POOL
                0084   391                                             ; STORED BY INIT CODE
00000000        0084   392         .LONG   0                           ; LAST + 1 VPN OF PAGED DYNAMIC POOL
                0088   393                                             ; STORED BY INIT CODE
30000000        0088   394         .LONG   <PTE$C_ERKW ! PTE$C_KOWN> ; DEMAND ZERO PTE
                008C   395 ;
                008C   396 ; SYSTEM WINDOW CONTROL BLOCK TEMPLATE FOR MAPPING THE SYSTEM IMAGE
                008C   397 ;
                008C   398 SYSWCB:
                008C   399         ASSUME  WCB$C_LENGTH EQ WCB$C_MAP
000000BC        008C   400 10$:     .BLKB   WCB$C_LENGTH                ; ALLOCATE THE FRONT OF A WINDOW
                00BC   401 20$:
00000094        00BC   402         .=10$+WCB$W_SIZE                    ; FILL IN THE SIZE FIELD
     0030       0094   403         .WORD   WCB$C_LENGTH                ; NOT COUNTING RETRIEVAL POINTERS
00000096        0096   404         .=10$+WCB$B_TYPE                    ; TYPE FIELD
          12    0096   405         .BYTE   DYN$C_WCB
00000097        0097   406         .=10$+WCB$B_ACCESS                  ; ACCESS FIELD
          65    0097   407         .BYTE   WCB$M_READ ! -              ; ACCESSED FOR READ
                0098   408                 WCB$M_NOTFCP ! -            ; NOT AN FCP WINDOW
                0098   409                 WCB$M_CATHEDRAL ! -         ; CATHEDRAL WINDOW
                0098   410                 WCB$M_COMPLETE              ; MAPS THE ENTIRE FILE
00000088        0098   411         .=10$+WCB$L_STVBN                   ; STARTING VIRTUAL BLOCK NUMBER
00000001        00B8   412         .LONG   1
000000BC        00BC   413         .=20$                               ; BACK TO END OF WCB
                00BC   414 ;
                00BC   415 ; HWTYPE TABLE
                00BC   416 ;
35 38 37 56     00BC   417         .ASCII  /V785/                      ; SPECIAL VARIANT OF 780
30 38 37 56     00C0   418 HWTYPE: .ASCII  /V780/                      ;
30 35 37 56     00C4   419         .ASCII  /V750/                      ;
30 33 37 56     00C8   420         .ASCII  /V730/                      ;
30 39 37 56     00CC   421         .ASCII  /V790/                      ;
53 53 38 56     00D0   422         .ASCII  /V8SS/                      ;
4E 4E 38 56     00D4   423         .ASCII  /V8NN/                      ;
31 56 55 56     00D8   424         .ASCII  /VUV1/                      ; MICRO-VAX 1
32 56 55 56     00DC   425         .ASCII  /VUV2/                      ;
                00E0   426 ;
                00E0   427 ; MESSAGES
                00E0   428 ;
                00E0   429 NOSPACE:                                     ; INSUFFICIENT NON-PAGED POOL
2D 54 49 4E 49 43 45 58 45 25 0A 0D  00E0   430         .ASCII  <CR><LF>/%EXECINIT-F-Insufficient non-paged pool/
65 69 63 69 66 66 75 73 6E 49 2D 46  00EC
64 65 67 61 70 2D 6E 6F 6E 20 74 6E  00F8
                  6C 6F 6F 70 20     0104
```

```
                                                00 0A 0D  0109   431          .ASCIZ  <CR><LF>                    ;
                                                          010C   432 NOPHYSMEM:                                 ;
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 010C   433          .ASCII  <CR><LF>/%EXECINIT-F-Insufficient physical memory for /
65 69 63 69 66 66 75 73 6E 49 2D 46             0000 0118
20 6C 61 63 69 73 79 68 70 20 74 6E             0000 0124
   20 72 6F 66 20 79 72 6F 6D 65 6D             0000 0130
6B 72 6F 77 20 6D 75 6D 69 6E 69 6D             0000 013B   434          .ASCIZ  /minimum working set/<CR><LF>
      00 0A 0D 74 65 73 20 67 6E 69             0000 0147
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 0151   435 NOSPT:   .ASCIZ  <CR><LF>/%EXECINIT-F-Insufficient SPT entries/<CR><LF>
65 69 63 69 66 66 75 73 6E 49 2D 46             0000 015D
69 72 74 6E 65 20 54 50 53 20 74 6E             0000 0169
                  00 0A 0D 73 65                0000 0175
                                                0000 017A   436 BADCONUCB:                                 ;
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 017A   437          .ASCIZ  <CR><LF>/%EXECINIT-F-Illegal OPA0: UCB size/<CR><LF>
50 4F 20 6C 61 67 65 6C 6C 49 2D 46             0000 0186
65 7A 69 73 20 42 43 55 20 3A 30 41             0000 0192
                                 00 0A 0D       0000 019E
                                                0000 01A1   438 BADDSKUCB:                                 ;
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 01A1   439          .ASCIZ  <CR><LF>/%EXECINIT-F-Illegal SYSDISK UCB size/<CR><LF>
59 53 20 6C 61 67 65 6C 6C 49 2D 46             0000 01AD
69 73 20 42 43 55 20 4B 53 49 44 53             0000 01B9
                  00 0A 0D 65 7A                0000 01C5
                                                0000 01CA   440 BADTTYDRV:                                 ;
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 01CA   441          .ASCIZ  <CR><LF>/%EXECINIT-F-Illegally formatted terminal service/<CR><LF>
20 79 6C 6C 61 67 65 6C 6C 49 2D 46             0000 01D6
65 74 20 64 65 74 74 61 6D 72 6F 66             0000 01E2
69 76 72 65 73 20 6C 61 6E 69 6D 72             0000 01EE
                  00 0A 0D 65 63                0000 01FA
                                                0000 01FF   442 BAD_ADDRESS:
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 01FF   443          .ASCII  <CR><LF>/%EXECINIT-F-Address is not within the/
73 69 20 73 73 65 72 64 64 41 2D 46             0000 020B
20 6E 69 68 74 69 77 20 74 6F 6E 20             0000 0217
                                 65 68 74       0000 0223
79 73 20 64 65 67 61 70 6E 6F 6E 20             0000 0226   444          .ASCIZ  / nonpaged system image, SYS.EXE/<CR><LF>
20 2C 65 67 61 6D 69 20 6D 65 74 73             0000 0232
      00 0A 0D 45 58 45 2E 53 59 53             0000 023E
                                                0000 0248   445 BAD_OPCODE:
2D 54 49 4E 49 43 45 58 45 25 0A 0D             0000 0248   446          .ASCII  <CR><LF>/%EXECINIT-F-Instruction mismatch between/
6F 69 74 63 75 72 74 73 6E 49 2D 46             0000 0254
62 20 68 63 74 61 6D 73 69 6D 20 6E             0000 0260
                  6E 65 65 77 74 65             0000 026C
20 65 64 6F 63 70 6F 20 64 6C 6F 20             0000 0272   447          .ASCIZ  / old opcode table and instruction stream/<CR><LF>
6E 69 20 64 6E 61 20 65 6C 62 61 74             0000 027E
74 73 20 6E 6F 69 74 63 75 72 74 73             0000 028A
                  00 0A 0D 6D 61 65 72          0000 0296
```

```
                                029D    449 ;+
                                029D    450 ; SYSTEM BOOT ENTRY POINTS TO START UP SYSTEM
                                029D    451 ;
                                029D    452 ; INPUTS:
                                029D    453 ;
                                029D    454 ;       R0 = PHYSICAL ADDRESS OF EXE$INIT
                                029D    455 ;       R11 = PHYSICAL ADDRESS OF RESTART PARAMETER BLOCK (RPB)
                                029D    456 ;       PR$_SBR/PR$_SLR - SET TO DESCRIBE SPT
                                029D    457 ;       PR$_P0BR/PR$_P0LR - SET TO MAP EXE$INIT VIRTUAL = REAL
                                029D    458 ;-
                                029D    459
                                029D    460         .ALIGN  LONG
                                02A0    461 EXE$INIT::                                  ; INIT START
                 5D    30 AB    D0      02A0    462         MOVL    RPB$L_BOOTR5(R11),FP    ; GET DEBUG FLAGS
                    38    01    DA      02A4    463         MTPR    #1,S^#PR$_MAPEN         ; ENABLE MAPPING
                000002AD'9F    17      02A7    464         JMP     @#10$                   ; AND SET PC IN SYSTEM SPACE
              5E    00000000'EF D0      02AD    465 10$:    MOVL    EXE$GL_INTSTK,SP                ; SET TO USE INTERRUPT STACK
00000000'EF     00000000'EF D0      02B4    466         MOVL    EXE$GL_DEFFLAGS,EXE$GL_FLAGS    ; ESTABLISH CORRECT DEFAULTS
              51    00000000'EF D0      02BF    467         MOVL    EXE$GL_SCB,R1           ; GET ADDRESS OF SCB
           28 A1    00000000'9F DE      02C6    468         MOVAL   @#EXE$TBIT,^X28(R1)     ; CONNECT SYS.EXE TRACE,
           2C A1    00000000'9F DE      02CE    469         MOVAL   @#EXE$BREAK,^X2C(R1)    ;  BREAKPOINT,
           18 A1    00000000'9F DE      02D6    470         MOVAL   @#EXE$ROPRAND,^X18(R1)  ;  RESERVED OPERAND,
           20 A1    00000000'9F DE      02DE    471         MOVAL   @#EXE$ACVIOLAT,^X20(R1) ;  ACCESS VIOLATION,
           24 A1    00000000'9F DE      02E6    472         MOVAL   @#MMG$PAGEFAULT,^X24(R1) ;  AND PAGE FAULT HANDLERS TO SCB
00000000'GF     000000F0 8F D3      02EE    473         BITL    #<ARC$M_CHAR_EMUL!ARC$M_DCML_EMUL!ARC$M_EDPC_EMUL! -
                                02F9    474                     ARC$M_CRC_EMUL>,G^EXE$GL_ARCHFLAG ; ARE STRING/DECIMAL/EDITPC/
                                02F9    475                                             ; CRC INSTRUCTIONS BEING EMULATED?
                    0B    13    02F9    476         BEQL    11$                     ; BR IF DONE IN HARDWARE/FIRMWARE
                54    0000'CF D0      02FB    477         MOVL    W^BOO$GL_VAXEMUL,R4     ; ADR WHERE EMULATION CODE IS LOADED
                  54    04 A4 C0      0300    478         ADDL2   4(R4),R4                ; GET ADR OF INITIALIZATION ROUTINE
                        64    16      0304    479         JSB     (R4)                    ; CONNECT CHAR EMUL TO SCB
                                0306    480 11$:
00000000'GF     00000F00 8F D3      0306    481         BITL    #<ARC$M_DFLT_EMUL!ARC$M_FFLT_EMUL!ARC$M_HFLT_EMUL! - ; ARE
                                0311    482                     ARC$M_GFLT_EMUL>,G^EXE$GL_ARCHFLAG ; FLT PT INS BEING EMULATED?
                    0B    13    0311    483         BEQL    12$                     ; BR IF DONE IN HARDWARE/FIRMWARE
                54    0000'CF D0      0313    484         MOVL    W^BOO$GL_FPEMUL,R4      ; ADR WHERE EMULATION CODE IS LOADED
                  54    04 A4 C0      0318    485         ADDL2   4(R4),R4                ; GET ADR OF INITIALIZATION ROUTINE
                        64    16      031C    486         JSB     (R4)                    ; CONNECT FLT PT EMUL TO OPCDEC IN SCB
                                031E    487 12$:
                        00'   E1      031E    488         BBC     S^#EXE$V_SSINHIBIT,-
                10 00000000'GF          0320    489                 G^EXE$GL_FLAGS,14$      ; IF WE ARE INHIBITING SYSTEM
           40 A1    00000000'GF DE      0326    490         MOVAL   G^EXE$CMODKRNLX,^X40(R1) ; SERVICES, REVECTOR THE ENTRY
           44 A1    00000000'GF DE      032E    491         MOVAL   G^EXE$CMODEXECX,^X44(R1) ; POINTS FOR THE CHMK/CHME SERVICES
                                0336    492 14$:
            52    51    15    09  EF      0336    493         EXTZV   #VA$V_VPN,#VA$S_VPN,R1,R2 ; GET THE VPN OF THE SCB
            52  00000000'FF42  D0      033B    494         MOVL    @MMG$GL_SPTBASE[R2],R2  ; GET PTE
            52    52    15    00  EF      0343    495         EXTZV   #PTE$V_PFN,#PTE$S_PFN,R2,R2     ; EXTRACT PFN
                  52    52    09  78      0348    496         ASHL    #9,R2,R2                ; AND CONVERT TO BYTE ADDRESS
                        11    52  DA      034C    497         MTPR    R2,S^#PR$_SCBB          ; SET SYSTEM CONTROL BLOCK BASE
                     14 5D    01  E0      034F    498         BBS     #RPB$V_DEBUG,FP,20$     ; KEEP DEBUGGER IF REQUESTED
                              01  01      0353    499 15$:    NOP                             ; SOURCE OF NOP OPCODE
00000000'EF     FC AF       90  0354    500         MOVB    15$,INI$BRK             ; PREVENT INITIAL BREAKPOINT
                FFC00000'8F D0      035C    501         MOVL    #<<XDELTABASE-^X80000000>@-9>,-
                    11E9'CF         0362    502                 W^FREE                  ; SET FREE DESCRIPTOR TO RELEASE DEBUGGER
                        46  11      0365    503         BRB     30$                     ; CONTINUE WITHOUT DEBUGGER
           2C A1    00000000'9F 9E      0367    504 20$:    MOVAB   @#XDELBPT,^X2C(R1)      ; SET VECTOR TO BPT
           28 A1    00000000'9F 9E      036F    505         MOVAB   @#XDELTBIT,^X28(R1)     ; SET TBIT VECTOR
```

```
                                0377   506 ; FOR LARGER THAN 32 MBYTES, USE LONGWORD FORMAT ...
                                0377   507         PFN_DISP_IF_BIGPFN_THEN         END_BIGPFN_CODE=23$
                                037F
                                037F                ;This code executes if the PFN link arrays are longword arrays.
00000000'GF   00000000'GF   DE  037F   508                 MOVAL   G^XDS$GT_LONG_PFN,G^XDS$GL_XESTRING ; SAVED XE STRING
00000000'GF   00000000'GF   DE  038A   509                 MOVAL   G^XDS$GT_LONG_PFN,G^XDS$GL_XFSTRING ; SAVED XF STRING
                                0395   510 ; OTHERWISE, USE WORD FORMAT
                                0395   511         PFN_DISP_ELSE           ELSE_CODE=23$,COMMON_CODE=26$
                                0397
                                0397                ;This code executes if the PFN link arrays are word arrays.
00000000'GF   00000000'GF   DE  0397   512                 MOVAL   G^XDS$GT_WORD_PFN,G^XDS$GL_XESTRING ; SAVED XE STRING
00000000'GF   00000000'GF   DE  03A2   513                 MOVAL   G^XDS$GT_WORD_PFN,G^XDS$GL_XFSTRING ; SAVED XF STRING
                                03AD   514         PFN_DISP_ENDIF          COMMON_CODE=26$
                                03AD
                                03AD                ;End of code that depends on size of PFN link arrays
                                03AD   515 30$:
                                03AD   516 ;
                                03AD   517 ; Load SYSLOAxxx and connect up the vectors so the console terminal I/O
                                03AD   518 ; routines can be used. The initialization routine for SYSLOA is called
                                03AD   519 ; later when the rest of the loadable code is initialized.
                                03AD   520 ;
                                03AD   521         INVALID                                 ; Clear temporary boot device mapping
                                03B0   522                                                 ; from translation buffer.
      52   0000'CF   D0  03B0   523         MOVL    W^BOO$GL_SYSLOA,R2      ; Address of SYSLOAxxx image in pool
00000000'EF   52   D0  03B5   524         MOVL    R2,XDEL_LOADBASE        ; Save base of loadable code in
                                03BC   525                                 ;   XDELTA X3 register
00000000'GF   52   D0  03BC   526         MOVL    R2,G^MMG$GL_SYSLOA_BASE ; store address of sysloa image in
                                03C3   527                                 ; in mmg field so that sda can find it.
53   00000000'GF   DE  03C3   528         MOVAL   G^EXE$AL_LOAVEC,R3      ; Address of resident vectors.
     00000000'GF   16  03CA   529         JSB     G^EXE$LINK_VEC          ; Connect SYSLOA vectors
     00000000'GF   16  03D0   530         JSB     G^CON$INIT_CTY          ; Initialize the console terminal
                   5B  D4  03D6   531
                   5B  D4  03D6   532         CLRL    R11                     ; INIDICATE CONSOLE TERMINAL
51   00000000'EF   DE  03D8   533         MOVAL   SYS$GT_ANNOUNCE,R1      ; GET ADDRESS OF ANNOUNCEMENT MESSAGE
     00000000'EF   16  03DF   534         JSB     EXE$OUTZSTRING          ; ANNOUNCE SYSTEM
                               03E5   535 ;
                               03E5   536 ; LOAD ARRAYS WITH VPN OF BOUNDARY BETWEEN NONPAGED AND PAGEABLE EXEC
                               03E5   537 ;
50 00000000'GF   15   09   EF  03E5   538         EXTZV   #VA$V_VPN,#VA$S_VPN,G^MMG$GL_PGDCOD,R0  ; MAKE ADDRESS INTO VPN
        FC81 CF   50   D0  03EE   539         MOVL    R0,PGDCOD               ; STORE IN PGDCOD ARRAY
        50   50   02   78  03F3   540         ASHL    #2,R0,R0                ; MAKE R0 INTO BYTE INDEX INTO SPT
00000046'EF   50   D0  03F7   541         MOVL    R0,PGDCOD_LIM           ; LOAD THIS VALUE INTO ARRAY USED
                               03FE   542                                 ;  BY INI$RDONLY/INI$WRITABLE
                   50   D7  03FE   543         DECL    R0                      ; (UPPER LIMIT IS ONE SMALLER
0000003A'EF   50   D0  0400   544         MOVL    R0,INI_RDONLY_LIST+4    ;  THAN LOWER LIMIT)
                               0407   545 ;
                               0407   546 ; SET UP NONPAGED POOL LISTHEAD AND INITIAL CONTENTS
                               0407   547 ;
50   00000000'EF   D0  0407   548         MOVL    MMG$GL_NPAGEDYN,R0      ; GET ADDRESS OF NON-PAGED POOL
                   60   D4  040E   549         CLRL    (R0)                    ; ZAP FORWARD LINK
 04 A0   0000'CF   D0  0410   550         MOVL    W^BOO$GL_NPAGEDYN,4(R0) ; SET SIZE OF FREE BLOCK
00000004'EF   50   D0  0416   551         MOVL    R0,EXE$GL_NONPAGED+4    ; SET ADDRESS OF POOL
                               041D   552
55   00000000'FF   DE  041D   553         MOVAL   @MMG$GL_SYSPHD,R5       ; GET ADDRESS OF SYSTEM HEADER
34 A5   00000000'EF   90  0424   554         MOVB    SGN$GB_SYSPFC,PHD$B_DFPFC(R5)   ; SET SYS PAGE FAULT CLUSTER
50   00000000'EF   DE  042C   555         MOVAL   EXE$GL_FLAGS,R0         ; PUT FLAGS ADDRESS IN CONVENIENT PLACE
      06 5D   02   E1  0433   556         BBC     #RPB$V_INIBPT,FP,NODEBUG ; BR IF NORMAL STARTUP(NO BREAKPOINT)
```

```
                                0437    557 ;
                                0437    558 ;       R0 = ADDRESS OF EXE$GL_FLAGS
                                0437    559 ;       R5 = ADDRESS OF SYSTEM PROCESS HEADER.
                                0437    560 ;
                00000000'EF   16 0437    561         JSB     INI$BRK                 ; OTHERWISE BREAKPOINT
                                043D    562 NODEBUG:                                ;
00000000'EF     00000000'EF   B0 043D    563         MOVW    SGN$GW_MAXPRCCT,SCH$GW_PROCLIM  ; SET TENATIVE LIMIT FOR PROCS
                                0448    564 ;
                                0448    565 ; SET MODIFIED PAGE WRITER PARAMETERS
                                0448    566 ;
00000000'EF     00000000'EF   3C 0448    567         MOVZWL  MPW$GW_HILIM,SCH$GL_MFYLIMSV ; LIST HIGH THRESHOLD
00000000'EF     00000000'EF   3C 0453    568         MOVZWL  MPW$GW_HILIM,SCH$GL_MFYLIM ; CURRENT AND SAVE VALUES
00000000'EF     00000000'EF   3C 045E    569         MOVZWL  MPW$GW_LOLIM,SCH$GL_MFYLOSV ; SAVE VALUE FOR LOW THRESHOLD
00000000'EF     00000000'EF   3C 0469    570         MOVZWL  MPW$GW_LOLIM,SCH$GL_MFYLOLIM ; LOW LIST THRESHOLD
                00000002'EF   16 0474    571         JSB     INI$WRITABLE            ; SYSTEM WRITABLE UNTIL INIT COMPLETES
                                047A    572
```

```
                        047A   574          .SUBTITLE       Alter PFN references if large PFN configuration
                        047A   575 ;+
                        047A   576 ;  If there is less than 32 Mbytes of memory described in the PFN data base
                        047A   577 ;  (MMG$GW_BIGPFN contains zero), this next block of code does nothing.
                        047A   578 ;  Otherwise, an address table is scanned.
                        047A   579 ;
                        047A   580 ;       1.  Each address must be in the nonpaged system image.
                        047A   581 ;
                        047A   582 ;       2.  The current contents are verified as a consistency check.
                        047A   583 ;
                        047A   584 ;       3.  A new (longword context) opcode is stored at that location.
                        047A   585 ;
                        047A   586 ;  Failure of either test prevents the system from being bootstrapped with
                        047A   587 ;  more than 32 Mbytes of physical memory. (That is, the PHYSICALPAGES
                        047A   588 ;  parameter must be used to allow the system to come up using less that
                        047A   589 ;  its total amount of physical memory.)
                        047A   590 ;-
                        047A   591
                        047A   592          PFN_DISP_IF_BIGPFN_THEN         END_BIGPFN_CODE=100$
                        0482
                        0482               ;This code executes if the PFN link arrays are longword arrays.
                        0482   593
    51  00000000'EF   DE    0482   594          MOVAL   MMG$AL_FIXUPTBL,R1      ; Address of opcode/address table
    52  00000000'EF   DE    0489   595          MOVAL   MMG$AL_ENDDRIVE,R2      ; SYS.EXE bounds check lower limit
    53  00000000'EF   D0    0490   596          MOVL    MMG$GL_PGDCOD,R3        ; SYS.EXE bounds check upper limit
        50    81      D0    0497   597 10$:     MOVL    (R1)+,R0                ; Get address of next fixup
                        049A   598                                             ;  (R1 now points to old opcode byte)
              29      13    049A   599          BEQL    50$                     ; Zero indicates end of list
        52    50      D1    049C   600          CMPL    R0,R2                   ; Is address too small?
              0F      1F    049F   601          BLSSU   20$                     ; Quit with error if too small
        53    50      D1    04A1   602          CMPL    R0,R3                   ; Is address too large?
              0A      1E    04A4   603          BGEQU   20$                     ; Error exit in this case, too
        60    81      91    04A6   604          CMPB    (R1)+,(R0)              ; Perform sanity check
                        04A9   605                                             ;  (R1 now points to new opcode byte)
              0C      12    04A9   606          BNEQU   30$                     ; Quit with error if different
        60    81      90    04AB   607          MOVB    (R1)+,(R0)              ; Finally, alter the opcode
                        04AE   608                                             ;  (R1 points to next address in table)
              E7      11    04AE   609          BRB     10$                     ;  and go back for the next one
                        04B0   610
    51  FD4B CF       9E    04B0   611 20$:     MOVAB   BAD_ADDRESS,R1          ; Select address-out-of-range message
              05      11    04B5   612          BRB     40$                     ;  and join common termination code
                        04B7   613
    51  FD8D CF       9E    04B7   614 30$:     MOVAB   BAD_OPCODE,R1           ; Select opcode-mismatch message
              5B      D4    04BC   615 40$:     CLRL    R11                     ; Specify console terminal
     00000000'EF      16    04BE   616          JSB     EXE$OUTZSTRING          ; Type out error message
                      00    04C4   617          HALT                            ;  and finally halt the processor
                        04C5   618 ;
                        04C5   619 ;  This is the successful exit path after all opcodes have been altered. It
                        04C5   620 ;  is necessary to execute an REI instruction in order that any instruction
                        04C5   621 ;  lookahead be invalidated and the new opcodes used.
                        04C5   622 ;
            7E      DC    04C5   623 50$:     MOVPSL  -(SP)                   ; Store PSL for REI
          CB'AF     9F    04C7   624          PUSHAB  B^100$                  ; Push PC also
                      02    04CA   625          REI                             ; Drop through to next instruction
                        04CB   626
                        04CB   627          PFN_DISP_ENDIF          COMMON_CODE=100$
                        04CB
```

```
                04CB                    ;End of code that depends on size of PFN link arrays
```

```
                                04CB    629          .SUBTITLE       Initialize pageable system code
                                04CB    630 INI_PAGING:
        56   00C0'CF    D0      04CB    631          MOVL    W^BOO$GL_BOOTCB,R6      ; ADDRESS OF BOOT CONTROL BLOCK
  00000000'EF   56    D0      04D0    632          MOVL    R6,EXE$GL_BOOTCB        ; SET ADDRESS IN SYSCOMMON
3E 00000000'EF   00'  E1      04D7    633          BBC     S^#EXE$V_SYSPAGING,EXE$GL_FLAGS,20$ ; BR IF NOT PAGING SYSTEM SPACE
       00000046'EF    7C      04DF    634          CLRQ    PGDCOD_LIM              ; PREVENT PROTECTION SETTING
    55  00000000'FF    DE      04E5    635          MOVAL   @MMG$GL_SYSPHD,R5       ; ADDRESS OF SYSTEM HEADER
         52   24 A5    32      04EC    636          CVTWL   PHD$W_PSTLAST(R5),R2    ; INDEX TO LAST SECTION TABLE ENTRY
              52   08    C2      04F0    637          SUBL    #SEC$C_LENGTH@-2,R2     ; ALLOCATED NEW SECTION TABLE ENTRY
           24 A5   52    B0      04F3    638          MOVW    R2,PHD$W_PSTLAST(R5)    ; UPDATE LAST ALLOCATED ENTRY
        FB80 CF    52    B0      04F7    639          MOVW    R2,PGDCOD+8             ; SET PAGED CODE SECTION INDEX
          55   20 A5    C0      04FC    640          ADDL    PHD$L_PSTBASOFF(R5),R5  ; BASE ADDRESS OF SECTION TABLE
            52   6542    DE      0500    641          MOVAL   (R5)[R2],R2             ; ADDRESS OF FIRST SECTION TABLE ENTRY
      10 A2   0C A6    D0      0504    642          MOVL    BOO$L_SYS_VBN(R6),SEC$L_VBN(R2) ; MAP THE ENTIRE SYSTEM IMAGE
              08 A2    D4      0509    643          CLRL    SEC$L_VPXPFC(R2)        ; STARTS AT SYS VIRTUAL PAGE 0
      0C A2   14 A6    D0      050C    644          MOVL    BOO$L_SYS_MAP(R6),SEC$L_WINDOW(R2) ; SYSTEM WINDOW
        18 A2   01    D0      0511    645          MOVL    #1,SEC$L_REFCNT(R2)     ; MAKE REFERENCE COUNT NON-ZERO
         55   FB5B CF    DE      0515    646          MOVAL   W^PGDCOD,R5             ; SET SFT FOR PAGED CODE
                0CA0    30      051A    647          BSBW    FILLSPT                 ; AND RELEASE THE PAGES OCCUPIED
                                051D    648 20$:                                     ;
                                051D    649 ;
                                051D    650 ;       PLACE ALL PAGES CURRENTLY REMAINING IN PFNMAP ON THE FREE LIST
                                051D    651 ;
                                051D    652 INI_FREEMEM:                             ;
    56   00000000'EF    D0      051D    653          MOVL    EXE$GL_RPB,R6           ; GET ADDRESS OF RPB
    50   FFC00000'8F    D0      0524    654          MOVL    #<<EXE$RESTART-^X80000000>@-9>,R0 ; VPN OF RESTART ROUTINE
 50   00000000'FF40    D0      052B    655          MOVL    @MMG$GL_SPTBASE[R0],R0  ; GET PTE FOR RESTART ROUTINE
  50   50   15   00    EF      0533    656          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R0,R0     ; EXTRACT PFN
    04 A6   50   09    78      0538    657          ASHL    #9,R0,RPB$L_RESTART(R6) ; SAVE PHYSICAL ADDRESS OF RESTART ROUTINE
              08 A6    D4      053D    658          CLRL    RPB$L_CHKSUM(R6)        ; INIT CHECKSUM ACCUMULATOR
              0C A6    D4      0540    659          CLRL    RPB$L_RSTRTFLG(R6)      ; ENABLE RESTART
    50   00000000'EF    9E      0543    660          MOVAB   EXE$RESTART,R0          ; ADDRESS OF RESTART ROUTINE
              51   1F    D0      054A    661          MOVL    #^X1F,R1                ; COUNT OF LONGWORDS IN CHECKSUM
            08 A6   80    C0      054D    662 5$:      ADDL    (R0)+,RPB$L_CHKSUM(R6)  ; ACCUMULATE CHECKSUM OF RESTART ROUTINE
                 F9 51    F5      0551    663          SOBGTR  R1,5$                   ; FOR ALL 31 LONGWORDS
         00AC C6    0C    DB      0554    664          MFPR    #PR$_SBR,RPB$L_SBR(R6)  ; SAVE SBR VALUE FOR RESTART
         00B8 C6    0D    DB      0559    665          MFPR    #PR$_SLR,RPB$L_SLR(R6)  ; SAVE SLR VALUE FOR RESTART
         00B0 C6    11    DB      055E    666          MFPR    #PR$_SCBB,RPB$L_SCBB(R6); AND SCB BASE ADDRESS
              00A4 C6    D4      0563    667          CLRL    RPB$L_ISP(R6)           ; CLEAR "SUCCESSFUL POWERFAIL" FLAG
      54   48 A6    17      0567    668          ROTL    #<32-9>,RPB$Q_PFNMAP+4(R6),R4 ; STARTING PFN OF BITMAP
     50   00000000'EF    D0      056C    669          MOVL    MMG$GL_MAXPFN,R0        ; START WITH HIGHEST PFN INCLUSIVE
      51   50   F4 8F    78      0573    670          ASHL    #-12,R0,R1              ; GET BITMAP PAGE NUMBER
              54   51    C0      0578    671          ADDL    R1,R4                   ; FIRST BITMAP PAGE TO MAP
     57   00000000'EF    D0      057B    672          MOVL    SWP$GL_BALBASE,R7       ; VA TO REFERENCE BITMAP PAGES
00000000'FF   54   B0000000 8F    C9      0582    673 10$:     BISL3   #<PTE$C_ERKW!PTE$M_VALID!PTE$C_KOWN>,R4,-
                                058E    674                  @SWP$GL_BALSPT          ; MAP A PAGE OF THE PFN BITMAP
                                058E    675          INVALID R7                      ; AND INVALIDATE THAT VA
         55   50   0C   00    EF      0591    676          EXTZV   #0,#12,R0,R5            ; BITMAP PAGE RELATIVE PFN
           06 67   55    E1      0596    677 20$:     BBC     R5,(R7),30$             ; BRANCH IF THIS PFN IS NOT USABLE
          00000000'EF    16      059A    678          JSB     MMG$DALLOCPFN           ; MAKE USABLE PFN AVAILABLE
                  50    D7      05A0    679 30$:     DECL    R0                      ; NEXT PFN TO CHECK
                  0F    13      05A2    680          BEQL    50$                     ; PFN ZERO MAY NOT BE USED
    00000000'EF   50    D1      05A4    681          CMPL    R0,MMG$GL_MINPFN        ; DONE THEM ALL?
                  06    19      05AB    682          BLSS    50$                     ; BRANCH IF YES
               E6 55    F4      05AD    683          SOBGEQ  R5,20$                  ; DO THE NEXT PAGE IN THIS BITMAP PAGE
               CF 54    F4      05B0    684          SOBGEQ  R4,10$                  ; DO THE NEXT BITMAP PAGE
                                05B3    685                                          ; THIS DOES NOT FALL THROUGH
```

```
                00000000'FF   D4  05B3   686 50$:    CLRL    @SWP$GL_BALSPT          ; CLEAN UP THE MAP ENTRY
                              05B9   687          INVALID R7                      ; AND THE TRANSLATION BUFFER
                              05BC   688
                              05BC   689 ;
                              05BC   690 ; INITIALIZE SPT FOR PAGED DYNAMIC POOL
                              05BC   691 ;
                              05BC   692 INI_PAGDYN:                              ;
54   00000000'EF   15   09   EF  05BC   693          EXTZV   #VA$V_VPN,#VA$S_VPN,MMG$GL_PAGEDYN,R4 ; VPN OF PAGED POOL
 50   00000000'EF   F7 8F   78  05C5   694          ASHL    #-9,SGN$GL_PAGEDYN,R0   ; OTHERWISE INIT IT AS NON-PAGED
           55   54   50     C1  05CE   695          ADDL3   R0,R4,R5                ; IF NOT PAGING POOL
 00000000'EF   00000000'EF   D0  05D2   696          MOVL    MMG$GL_PAGEDYN,EXE$GL_PAGED     ; SET ADDRESS OF PAGED POOL
      30 00000000'EF   00'   E0  05DD   697          BBS     S^#EXE$V_POOLPGING,EXE$GL_FLAGS,30$ ; BRANCH IF PAGING PAGED POOL
       50   00000000'FF44   D0  05E5   698 10$:     MOVL    @MMG$GL_SPTBASE[R4],R0  ; SPT ENTRY
            00000000'EF     16  05ED   699          JSB     MMG$ALLOCPFN            ; OTHEWISE GET A PFN
             01 50     1F   E1  05F3   700          BBC     #31,R0,20$              ; BRANCH IF GOT ONE
                       00   05F7   701          HALT                            ; NO PAGES FOR POOL, DISASTER!!
          50   FFE00000 8F   CA  05F8   702 20$:     BICL    #^C<PTE$M_PFN>,R0       ; LEAVE ONLY PFN BITS
            B0000000 8F   C9  05FF   703          BISL3   #<PTE$M_VALID ! PTE$C_ERKW ! PTE$C_KOWN>,-
       00000000'FF44   50         0605   704                  R0,@MMG$GL_SPTBASE[R4]  ; SET NEW PAGE TABLE ENTRY
                 0BE6   30   060C   705          BSBW    SETRESIDENT             ; SET THE PFN RESIDENT
             D2 54   55   F2  060F   706          AOBLSS  R5,R4,10$               ; FOR EACH PAGE IN THE POOL
                    19   11  0613   707          BRB     40$                     ;
                         0615   708
                         0615   709 ;
                         0615   710 ; SET UP ADDRESS OF PAGED POOL AND INIT IT FOR PAGING IF ENABLED
                         0615   711 ;
           55   FA67 CF   DE  0615   712 30$:     MOVAL   W^PAGEDYN,R5            ; ADDRESS OF PAGED POOL DESCRIPTORS
                65   54   D0  061A   713          MOVL    R4,(R5)                 ; 1ST VPN OF PAGED POOL
 7E   00000000'EF   F7 8F   78  061D   714          ASHL    #-9,SGN$GL_PAGEDYN,-(SP); GET SIZE OF PAGED POOL IN PAGES
       04 A5   54   8E   C1  0626   715          ADDL3   (SP)+,R4,4(R5)           ; LAST + 1 OF PAGED POOL
                 0B8F   30   062B   716          BSBW    FILLSPT                 ; SET SPT FOR PAGED POOL
                         062E   717 40$:                                         ;
                         062E   718 ;
                         062E   719 ; INITIALIZE LOOKASIDE I/O PACKET POOL
                         062E   720 ;
                         062E   721 INI_IRP:                                     ;
          50   0000'CF   7D  062E   722          MOVQ    W^BOO$GL_SPLITADR,R0    ; R0 = BASE ADDRESS OF IRP LIST
                         0633   723                                          ; R1 = NO. OF IRP'S TO INITIALIZE
       00000000'EF   50   D0  0633   724          MOVL    R0,EXE$GL_SPLITADR      ; SET LOOKASIDE LIST SPLIT ADDRESS
       00000000'EF   51   D0  063A   725          MOVL    R1,IOC$GL_IRPCNT        ; SET CURRENT COUNT OF IRPS
                     2F   13  0641   726          BEQL    INI_LRP                 ; SKIP IF NONE
                     14   11  0643   727          BRB     140$                    ; 0 OR MORE TRIPS THROUGH THE LOOP
       00000000'FF   60   0E  0645   728 130$:    INSQUE  (R0),@IOC$GL_IRPBL      ; INSERT I/O PACKET IN LOOKASIDE LIST
        08 A0   00D0 8F   B0  064C   729          MOVW    #<IRP$C_LENGTH+^XF>&<^C<^XF>>,IRP$W_SIZE(R0) ; SET SIZE
       50   000000D0 8F   C0  0652   730          ADDL    #<IRP$C_LENGTH+^XF>&<^C<^XF>>,R0 ; ADVANCE TO NEXT I/O PACKET
                E9 51   F4  0659   731 140$:    SOBGEQ  R1,130$                 ;
      52   50   09   00   EF  065C   732          EXTZV   #0,#9,R0,R2             ; GET OFFSET IN PAGE
                     0F   13  0661   733          BEQL    INI_LRP                 ; IF ZERO, NO PARTIAL PACKET
60   00000200 8F   52   C3  0663   734          SUBL3   R2,#512,(R0)            ; SAVE SIZE OF PARTIAL PACKET IN
                         066B   735                                          ;  FIRST LONGWORD OF FRAGMENT
       00000000'EF   50   D0  066B   736          MOVL    R0,IOC$GL_IRPREM        ; SAVE ADDRESS OF PARTIAL PACKET
                         0672   737 ;
                         0672   738 ; INITIALIZE LARGE REQUEST PACKET LOOK ASIDE LIST
                         0672   739 ;
                         0672   740 INI_LRP:
          52   0000'CF   D0  0672   741          MOVL    W^BOO$GL_LRPSIZE,R2     ; LRP SIZE FROM SYSBOOT
       00000000'EF   52   D0  0677   742          MOVL    R2,IOC$GL_LRPSIZE       ; STORE LRP SIZE
```

```
00000000'EF   0000'CF   D0   067E   743          MOVL    W^BOO$GL_LRPMIN,IOC$GL_LRPMIN ; STORE LRP MINIMUM SIZE
     00000000'EF   52   D1   0687   744          CMPL    R2,IOC$GL_LRPMIN          ; MAKE SURE MIN < SIZE
                   07   1E   068E   745          BGEQU   10$
     00000000'EF   52   D0   0690   746          MOVL    R2,IOC$GL_LRPMIN          ; USE MIN = SIZE
          50   0000'CF   7D   0697   747 10$:     MOVQ    W^BOO$GL_LRPSPLIT,R0      ; R0 = LRP LOOKASIDE LIST SPLIT ADR
                          069C   748                                               ; R1 = NO. OF LRP'S TO INITIALIZE
     00000000'EF   50   D0   069C   749          MOVL    R0,IOC$GL_LRPSPLIT        ; SET LOOKASIDE LIST SPLIT ADDRESS
     00000000'EF   51   D0   06A3   750          MOVL    R1,IOC$GL_LRPCNT          ; SAVE CURRENT LRP COUNT
                   29   13   06AA   751          BEQL    50$                       ; SKIP IF NONE
                   0E   11   06AC   752          BRB     40$                       ; 0 OR MORE TRIPS THROUGH LOOP
     00000000'FF   60   0E   06AE   753 30$:     INSQUE  (R0),@IOC$GL_LRPBL        ; INSERT I/O PACKET IN LOOKASIDE LIST
           08 A0   52   B0   06B5   754          MOVW    R2,IRP$W_SIZE(R0)         ; SET SIZE
              50   52   C0   06B9   755          ADDL    R2,R0                     ; ADVANCE TO NEXT I/O PACKET
                 EF 51   F4   06BC   756 40$:     SOBGEQ  R1,30$                    ;
     51   50   09   00   EF   06BF   757          EXTZV   #0,#9,R0,R1               ; GET OFFSET IN PAGE
                   0F   13   06C4   758          BEQL    50$                       ; IF ZERO, NO PARTIAL PACKET
60   00000200 8F   51   C3   06C6   759          SUBL3   R1,#512,(R0)              ; SAVE SIZE OF PARTIAL PACKET IN
                          06CE   760                                               ;  FIRST LONGWORD.
     00000000'EF   50   D0   06CE   761          MOVL    R0,IOC$GL_LRPREM          ; SAVE ADDRESS OF PARTIAL PACKET
                          06D5   762 50$:                                          ;
                          06D5   763 ;
                          06D5   764 ; INITIALIZE SMALL REQUEST PACKET LOOK ASIDE LIST
                          06D5   765 ;
                          06D5   766 INI_SRP:
     52   00000000'EF   D0   06D5   767          MOVL    SGN$GL_SRPSIZE,R2         ; SRP SIZE FROM SYSBOOT
              52   0F   C0   06DC   768          ADDL    #15,R2                    ; ROUND UP TO 16 BYTE BOUNDARY
              52   0F   CA   06DF   769          BICL    #15,R2
     00000000'EF   52   D0   06E2   770          MOVL    R2,IOC$GL_SRPSIZE         ; STORE SRP SIZE
00000000'EF   00000000'EF   D0   06E9   771          MOVL    SGN$GL_SRPMIN,IOC$GL_SRPMIN ; ** NOTE MEMORYALC MAY IGNORE THIS
     00000000'EF   52   D1   06F4   772          CMPL    R2,IOC$GL_SRPMIN          ; MAKE SURE MIN < SIZE
                   07   1E   06FB   773          BGEQU   10$
     00000000'EF   52   D0   06FD   774          MOVL    R2,IOC$GL_SRPMIN          ; USE MIN = SIZE
          50   00C0'CF   D0   0704   775 10$:     MOVL    W^BOO$GL_SRPSPLIT,R0      ; R0 = SRP LOOKASIDE LIST SPLIT ADR
     51   00000000'EF   D0   0709   776          MOVL    SGN$GL_SRPCNT,R1          ; R1 = NO. OF SRP'S TO INITIALIZE
     00000000'EF   50   D0   0710   777          MOVL    R0,IOC$GL_SRPSPLIT        ; SET LOOKASIDE LIST SPLIT ADDRESS
     00000000'EF   51   D0   0717   778          MOVL    R1,IOC$GL_SRPCNT          ; SAVE CURRENT SRP COUNT
                   29   13   071E   779          BEQL    50$                       ; SKIP IF NONE
                   0E   11   0720   780          BRB     40$                       ; 0 OR MORE TRIPS THROUGH LOOP
     00000000'FF   60   0E   0722   781 30$:     INSQUE  (R0),@IOC$GL_SRPBL        ; INSERT I/O PACKET IN LOOKASIDE LIST
           08 A0   52   B0   0729   782          MOVW    R2,IRP$W_SIZE(R0)         ; SET SIZE
              50   52   C0   072D   783          ADDL    R2,R0                     ; ADVANCE TO NEXT I/O PACKET
                 EF 51   F4   0730   784 40$:     SOBGEQ  R1,30$                    ;
     51   50   09   00   EF   0733   785          EXTZV   #0,#9,R0,R1               ; GET OFFSET IN PAGE
                   0F   13   0738   786          BEQL    50$                       ; IF ZERO, NO PARTIAL PACKET
60   00000200 8F   51   C3   073A   787          SUBL3   R1,#512,(R0)              ; SAVE SIZE OF PARTIAL PACKET IN
                          0742   788                                               ;  FIRST LONGWORD.
     00000000'EF   50   D0   0742   789          MOVL    R0,IOC$GL_SRPREM          ; SAVE ADDRESS OF PARTIAL PACKET
                          0749   790 50$:                                          ;
                          0749   791
00000000'EF   00000000'EF   D0   0749   792          MOVL    IOC$GL_SRPSIZE,IOC$GL_IRPMIN ; SET MIN SIZE
              00000000'EF   D6   0754   793          INCL    IOC$GL_IRPMIN             ; DON'T LEAVE A HOLE AFTER SRP SIZE
     5B   00000000'EF   D0   075A   794          MOVL    EXE$GL_NONPAGED,R11       ; Save IPL for pool allocation.
                   12   DB   0761   795          MFPR    S^#PR$_IPL,-              ; Set it to 31 for allocations
              00000000'EF        0763   796                   EXE$GL_NONPAGED      ; during INIT execution.
                          0768   797 ;
                          0768   798 ; SET UP FILEREAD GLOBAL PARAMETERS PASSED FROM SYSBOOT
                          0768   799 ;
```

```
        00000000'EF    0000'CF    7D  0768    800          MOVQ    W^BOO$GQ_FILCACHE,FIL$GQ_CACHE ; SET UP CACHE
    00000000'EF    0000'CF    0A   28  0771    801          MOVC3   #10,W^BOO$GT_TOPSYS,FIL$GT_TOPSYS ; SET TOP LEVEL SYSTEM DIR
                                   077B    802 ;
                                   077B    803 ; NO ALLOCATION OF NON-PAGED POOL BEFORE THIS POINT !!!!
                                   077B    804 ;
```

```
                                 077B   806          .SBTTL  Miscellaneous Initialization
                                 077B   807 ;
                                 077B   808 ; Initialize the permanent local system block.
                                 077B   809 ;
      55   00000000'9F    DE     077B   810          MOVAL   @#SCS$GA_LOCALSB,R5             ; Cover the system block
   18 A5   00000000'9F    7D     0782   811          MOVQ    @#SCS$GB_SYSTEMID, -
                                 078A   812                    SB$B_SYSTEMID(R5)          ; Set in the systemid
                1E A5    B4     078A   813          CLRW    SB$B_SYSTEMID+6(R5)            ; Just in case
   24 A5   20534D56 8F    D0     078D   814          MOVL    #^A/VMS /,SB$T_SWTYPE(R5)      ; Set operating system name
   28 A5   00000000'8F    D0     0795   815          MOVL    #SYS$K_VERSION,SB$T_SWVERS(R5) ; Set operating system version
                                 079D   816 ;
                                 079D   817 ; Set NODE_HWTYPE value based upon the CPUTYPE value. This code is modeled
                                 079D   818 ; on the CPUDISP macro.
                                 079D   819 ;
      50   00000000'GF    9A     079D   820          MOVZBL  G^EXE$GB_CPUTYPE,R0            ; Get CPU type and use as index
                  0A 50    F5     07A4   821          SOBGTR  R0,60$                         ; If GTR, then type <> 780
   02 00000000'GF   17    E1     07A7   822          BBC     #23,G^EXE$GB_CPUDATA,60$       ; If bit clear, then not a 785
                    50    D7     07AF   823          DECL    R0                             ; Set up as 785
     34 A5    F90A CF40    D0     07B1   824 60$:     MOVL    HWTYPE[R0],SB$T_HWTYPE(R5)     ; Store CPU type string in SB
                                 07B8   825
   38 A5   00000000'GF    7D     07B8   826          MOVQ    G^EXE$GB_CPUDATA,SB$B_HWVERS(R5); Copy CPU data (hardware/ ucode
00000000'9F    08    20     3A     07C0   827          LOCC    #^A/ /,#8,@#SCS$GB_NODENAME    ; Find the end of the name
         50    08    50     C3     07C8   828          SUBL3   R0,#8,R0                      ; Calculate name size
      44 A5    50    90     07CC   829          MOVB    R0,SB$T_NODENAME(R5)           ; Insert size
      00000000'9F    7D     07D0   830          MOVQ    @#EXE$GQ_TODCBASE,-            ; Copy the last boot time
                  2C A5           07D6   831                    SB$Q_SWINCARN(R5)          ;  as the incarnation number
OF    00   00000000'9F    50    2C     07D8   832          MOVC5   R0,@#SCS$GB_NODENAME,#0,#15, -
                  45 A5           07E1
                                 07E3   833                    SB$T_NODENAME+1(R5)        ; Insert the name
```

```
                        07E3    835          .SBTTL  Connect up loadable CPU-dependent code (SYSLOAxxx.EXE)
                        07E3    836 ;
                        07E3    837 ; The loadable CPU-dependent image (SYSLOAxxx.EXE) is now allocated
                        07E3    838 ; non-paged pool space and read into it by SYSBOOT.  The address of this
                        07E3    839 ; code is passed in BOO$GL_SYSLOA.  Link the resident system vectors at
                        07E3    840 ; EXE$AL_LOAVEC.  The loadable file now has self-describing vector and
                        07E3    841 ; offset information within it.
                        07E3    842 ;
                        07E3    843 ; The SYSLOAxxx.EXE image starts with a longword containing the load image
                        07E3    844 ; size, a longword of zero, a longword of standard pool header, followed
                        07E3    845 ; by a list of self describing entries.  Each entry consists of a type byte
                        07E3    846 ; and a longword self-relative offset to the loaded subroutine.
                        07E3    847 ;
                        07E3    848 INI_LOADCODE:
                        07E3    849          INVALID                                ; Clear temporary boot device mapping
                        07E6    850                                                 ; from translation buffer.
     54  0000'CF   DO   07E6    851          MOVL    W^BOO$GL_SYSLOA,R4             ; Address of SYSLOAxxx image in pool
           54   10   07EB    852          BSBB    LINK_INIT_RTN                  ; Call initialization routine
                        07ED    853                                                 ; SYSLOAxxx has already been connected
                        07ED    854
                        07ED    855
                        07ED    856 ;
                        07ED    857 ; Connect up loadable $ERAPAT code.  The loadable $ERAPAT code image
                        07ED    858 ; (ERAPATLOA.EXE) has been loaded into pool (if necessary) by SYSBOOT.
                        07ED    859 ; The address of the code is passed in BOO$GL_ERAPATLOA.  Link the system
                        07ED    860 ; vectors (at EXE$ERAPAT_VEC) to the loaded code if the code was loaded.
                        07ED    861 ;
                        07ED    862
                        07ED    863 ERAPAT_LOADCODE:
     52  0000'CF   DO   07ED    864          MOVL    W^BOO$GL_ERAPATLOA,R2          ; Get address in pool of loaded code
           09   13   07F2    865          BEQL    10$                            ; It wasn't loaded
 53  00000000'GF   DE   07F4    866          MOVAL   G^EXE$ERAPAT_VEC,R3            ; Get address of vectors in SYS.EXE
           3B   10   07FB    867          BSBB    LINK_INIT                      ; Connect vectors to loaded routines
                        07FD    868 10$:
                        07FD    869
                        07FD    870 ; Connect up loadable $MTACCESS code.  The loadable $MTACCESS code image
                        07FD    871 ; (MTACCESS.EXE) has been loaded into pool (if necessary) by SYSBOOT.
                        07FD    872 ; The address of the code is passed in BOO$GL_MTACCESSLOA.  Link the system
                        07FD    873 ; vectors (at EXE$MTACCESS_VEC) to the loaded code if the code was loaded.
                        07FD    874
                        07FD    875 MTACCESS_LOADCODE:
     52  0000'CF   DO   07FD    876          MOVL    W^BOO$GL_MTACCESSLOA,R2 ; Get address in pool of loaded code
           09   13   0802    877          BEQL    10$                            ; It wasn't loaded
 53  00000000'GF   DE   0804    878          MOVAL   G^EXE$MTACCESS_VEC,R3          ; Get address of vectors in SYS.EXE
           2B   10   080B    879          BSBB    LINK_INIT                      ; Connect vectors to loaded routines
                        080D    880 10$:
```

```
                     080D   882          .SBTTL  Connect up loadable SCS code (SCSLOA.EXE)
                     080D   883 ;
                     080D   884 ; The loadable SCS code image (SCSLOA.EXE) is now allocated non-paged
                     080D   885 ; pool space and read into it by SYSBOOT.  The address of this code
                     080D   886 ; is passed in BOO$GL_SCSLOA.  Link the resident system vectors at
                     080D   887 ; SCS$AL_LOAVEC.  The loadable file has self-describing vector and
                     080D   888 ; offset information within it.
                     080D   889 ;
                     080D   890 SCS_LOADCODE:
        0000'CF   D0 080D   891          MOVL    W^BOO$GL_UCODE,-          ; Transfer the address of any
 00000000'GF       0811   892                  G^SCS$GL_MCADR            ;  loaded microcode
    52   0000'CF   D0 0816   893          MOVL    W^BOO$GL_SCSLOA,R2        ; Address of SCSLOA image in pool
              09   13 081B   894          BEQL    10$                       ; Not loaded
53   00000000'GF   DE 081D   895          MOVAL   G^SCS$AL_LOAVEC,R3        ; Address of resident vectors.
              12   10 0824   896          BSBB    LINK_INIT                 ; Connect vectors to loaded routines
                     0826   897 10$:
                     0826   898
                     0826   899
                     0826   900 ;
                     0826   901 ; Connect up loadable cluster code.  The loadable cluster code image
                     0826   902 ; (CLUSTRLOA.EXE) has been loaded into pool (if necessary) by SYSBOOT.
                     0826   903 ; The address of the code is passed in BOO$GL_CLSLOA.  Link the system
                     0826   904 ; vectors (at CLU$AL_LOAVEC) to the loaded code if the code was loaded.
                     0826   905 ;
                     0826   906 ; This must be initialized AFTER the SCS loadable code.
                     0826   907 ; This must be initialized after and after the local system block.
                     0826   908 ;
                     0826   909 CLU_LOADCODE:
    52   0000'CF   D0 0826   910          MOVL    W^BOO$GL_CLSLOA,R2        ; Get address in pool of loaded code
              09   13 082B   911          BEQL    10$                       ; It wasn't loaded
53   00000000'GF   DE 082D   912          MOVAL   G^CLU$AL_LOAVEC,R3        ; Get address of vectors in SYS.EXE
              02   10 0834   913          BSBB    LINK_INIT                 ; Connect vectors to loaded routines
                     0836   914 10$:
              1A   11 0836   915          BRB     END_LOA
                     0838   916
                     0838   917 LINK_INIT:
    54   52   D0 0838   918          MOVL    R2,R4                     ; Save
00000000'GF   16 083B   919          JSB     G^EXE$LINK_VEC            ; Connect vectors to loaded routines.
                     0841   920 LINK_INIT_RTN:
 50   04 A4   D0 0841   921          MOVL    4(R4),R0                  ; Possible initialization routine
           0A   13 0845   922          BEQL    9$                        ; None, leave
         6044   16 0847   923          JSB     (R0)[R4]                  ; Call it
        C4 50   E8 084A   924          BLBS    R0,9$                     ; No errors
         0A0D   30 084D   925          BSBW    NOPOOLERR                 ; Trouble! Not enough memory
                00 0850   926          HALT                              ; **** FATAL ERROR ****
                     0851   927
                05   0851   928 9$:     RSB
                     0852   929
                     0852   930 END_LOA:
```

```
                                0852   932          .SBTTL  Initialize real time SPT bit map
                                0852   933 ;
                                0852   934 ; Allocate and initialize a bit map that describes the SPTs reserved
                                0852   935 ; via a SYSBOOT parameter for use by real time processes that issue
                                0852   936 ; connect to interrupt requests.
                                0852   937 ;
                                0852   938 INI_SPT:
                                0852   939 ;
                                0852   940 ; See if the number of real time SPTs requested by the system
                                0852   941 ; parameter is available in the SPT free list.
                                0852   942 ;
      55   00000000'GF   D0   0852   943          MOVL    G^EXE$GL_RTIMESPT,R5    ; Get number requested.
                     03   12   0859   944          BNEQ    5$                      ; If any, branch and proceed.
                   0079   31   085B   945          BRW     END_INISPT              ; in system initialization.
                                085E   946
56    55   00000000'GF   C1   085E   947 5$:       ADDL3   G^BOO$GL_SPTFREL,R5,R6  ; Add to base of free SPTs.
      00000000'GF   56   D1   0866   948          CMPL    R6,G^BOO$GL_SPTFREH     ; Are there enough left?
                     0E   15   086D   949          BLEQ    10$                     ; Yes. Branch forward.
          51   F8DE   CF   9E   086F   950          MOVAB   NOSPT,R1                ; No. Report error.
                     5B   D4   0874   951          CLRL    R11                     ; Specify console terminal
           00000000'EF   16   0876   952          JSB     EXE$OUTZSTRING          ; Output error report.
                     00   087C   953          HALT                            ; And halt processor.
                                087D   954 ;
                                087D   955 ; Calculate size of bit map control block needed and allocate it.
                                087D   956 ;
       51    55    FB 8F   78   087D   957 10$:      ASHL    #-5,R5,R1               ; Calculate number of longwords
                                0882   958                                   ; needed for bit map.
             55    1F   D3   0882   959          BITL    #^X1F,R5                ; Need to round up?
                     02   13   0885   960          BEQL    20$                     ; No. Branch forward.
                     51   D6   0887   961          INCL    R1                      ; Yes. Add one more longword.
     51      51    02   78   0889   962 20$:      ASHL    #2,R1,R1                ; Convert to byte count.
             51    0C   C0   088D   963          ADDL    #RBM$K_LENGTH,R1        ; R1 = realtime bitmap block size.
                 0987   30   0890   964          BSBW    ALONONPAGED             ; Allocate from nonpaged pool.
                                0893   965                                   ; ALONONPAGED halts on error.
00000000'GF      52   D0   0893   966          MOVL    R2,G^EXE$GL_RTBITMAP    ; Save block address.
             57  52   D0   089A   967          MOVL    R2,R7                   ; Get another copy of block address.
                                089D   968 ;
                                089D   969 ; Translate starting offset of starting SPT to the system virtual
                                089D   970 ; address of the page table entry. Then initialize the control block.
                                089D   971 ;
                                089D   972          ASSUME  RBM$L_STARTVPN EQ 0
87   00000000'GF      D0   089D   973          MOVL    G^BOO$GL_SPTFREL,(R7)+  ; Store starting virtual page
                                08A4   974                                   ; number in control block.
00000000'GF      56   D0   08A4   975          MOVL    R6,G^BOO$GL_SPTFREL     ; Save new first free SPT.
                                08AB   976          ASSUME  RBM$L_FREECOUNT EQ RBM$L_STARTVPN+4
             87  55   D0   08AB   977          MOVL    R5,(R7)+                ; Store number of SPTs.
                                08AE   978          ASSUME  RBM$W_SIZE EQ RBM$L_FREECOUNT+4
             87  51   B0   08AE   979          MOVW    R1,(R7)+                ; Set block size.
                                08B1   980          ASSUME  RBM$B_TYPE EQ RBM$W_SIZE+2
             87  31   B0   08B1   981          MOVW    #DYN$C_RBM,(R7)+        ; Store block type.
                                08B4   982          ASSUME  RBM$L_BITMAP EQ RBM$B_TYPE+2
                                08B4   983 ;
                                08B4   984 ; In the bit map section of the control block, set each bit that
                                08B4   985 ; corresponds to a reserved SPT.
                                08B4   986 ;
                     56   D4   08B4   987          CLRL    R6                      ; Starting bit number is zero.
             55  20   D1   08B6   988 30$:      CMPL    #32,R5                  ; More than a longword of bits to set?
```

```
                          12  18  08B9   989          BGEQ    40$                     ; No. Do last longword.
        20  56  FFFFFFFF  8F  F0  08BB   990          INSV    #-1,R6,#32,-            ; Set a longword worth of bits.
                      0C  A2      08C3   991                  RBM$L_BITMAP(R2)
                    56    20  C0  08C5   992          ADDL    #32,R6                  ; Move to next longword.
                    55    20  C2  08C8   993          SUBL    #32,R5                  ; Decrement count by bits set.
                          E9  11  08CB   994          BRB     30$                     ; Go alter more bits.
                              08CD   995
 0C A2  55  56  FFFFFFFF  8F  F0  08CD   996 40$:     INSV    #-1,R6,R5,-             ; Set remaining bits.
                              08D7   997                  RBM$L_BITMAP(R2)
                              08D7   998 END_INISPT:
```

```
                                08D7  1000          .SBTTL  Initialize Lock Manager Data Structures
                                08D7  1001 ;
                                08D7  1002 ; ALLOCATE AND INITIALIZE THE LOCK ID TABLE, THE RESOURCE HASH TABLE AND
                                08D7  1003 ; THE PROCESS BITMAP.  THE LOCK ID TABLE IS INITIALIZED WITH EACH LONGWORD
                                08D7  1004 ; CONTAINING THE INDEX OF THE NEXT LONGWORD.  THE RESOURCE HASH TABLE
                                08D7  1005 ; IS INITIALIZED TO ZERO.  THE PROCESS BITMAP DOES NOT HAVE TO BE
                                08D7  1006 ; INITIALIZED.
                                08D7  1007 ;
                                08D7  1008 INI_LCKIDTBL:
    51   00000000'EF    04    C5  08D7  1009          MULL3   #4,LCK$GL_IDTBLSIZ,R1   ; MULTIPLY NUMBER OF ENTRIES BY 4
                  51    0C    C0  08DF  1010          ADDL    #12,R1                  ; AND ALLOCATE THAT SIZE + 12
                     0935    30  08E2  1011          BSBW    ALONONPAGED             ; BYTES OF PREFIX
                  62    51    D0  08E5  1012          MOVL    R1,0(R2)                ; STORE SIZE ALLOCATED
                    02 A2    B5  08E8  1013          TSTW    2(R2)                   ; IS IT BIGGER THAN 65K?
                       04    12  08EB  1014          BNEQ    5$                      ; YES
              08 A2    51    B0  08ED  1015          MOVW    R1,8(R2)                ; NO, STORE SIZE IN SIZE FIELD
              0A A2    37    90  08F1  1016 5$:      MOVB    #DYN$C_LKID,10(R2)      ; STORE STRUCTURE TYPE
    00000000'EF    0C A2    9E  08F5  1017          MOVAB   12(R2),LCK$GL_IDTBL     ; STORE ADDRESS OF START OF TABLE
                  52    10    C0  08FD  1018          ADDL    #16,R2                  ; POINT TO SECOND ENTRY IN TABLE
                  51    10    C2  0900  1019          SUBL    #16,R1                  ; COMPUTE NUMBER OF ENTRIES
                  51    04    C6  0903  1020          DIVL    #4,R1                   ; LESS ONE
        00000000'EF    51    D0  0906  1021          MOVL    R1,LCK$GL_MAXID         ; STORE MAX LOCK ID
        00000000'EF    01    D0  090D  1022          MOVL    #1,LCK$GL_NXTID         ; INITIALIZE NEXT ID TO 1
                  53    02    D0  0914  1023          MOVL    #2,R3                   ; SETUP TO INITIALIZE REST OF TABLE
                  82    53    B0  0917  1024 10$:     MOVW    R3,(R2)+                ; STORE INDEX OF NEXT ENTRY IN THIS ENTRY
                  82    01    B0  091A  1025          MOVW    #1,(R2)+                ; STORE SEQUENCE NUMBER
               F6 53    51    F3  091D  1026          AOBLEQ  R1,R3,10$               ; REPEAT FOR ALL ENTRIES EXCEPT THE LAST
        82    00010000 8F    D0  0921  1027          MOVL    #^X10000,(R2)+          ; STORE LAST ENTRY
                                0928  1028 ;
                                0928  1029 ; ALLOCATE RESOURCE HASH TABLE.  THE NUMBER OF ENTRIES IN THE HASH TABLE
                                0928  1030 ; MUST BE A POWER OF TWO.  SO THE ALLOCATED SIZE IS THE SMALLEST POWER OF
                                0928  1031 ; TWO LARGER THAN THE SYSGEN PARAMETER.
                                0928  1032 ;
                                0928  1033 INI_RESHTBL:
                  51    01    D0  0928  1034          MOVL    #1,R1                   ; SMALLEST POSSIBLE HASH TABLE IS 1 ENTRY
                  54    D4  092B  1035          CLRL    R4                      ; R4 WILL BE POWER OF TWO ENTRIES
        00000000'EF    51    D1  092D  1036 10$:     CMPL    R1,LCK$GL_HTBLSIZ       ; IS R1 >= SPECIFIED SIZE?
                  07    1E  0934  1037          BGEQU   20$                     ; YES
                  51    02    C4  0936  1038          MULL    #2,R1                   ; NO - MULTIPLY SIZE BY TWO
                  54    D6  0939  1039          INCL    R4                      ; INCREMENT POWER OF TWO
                  F0    11  093B  1040          BRB     10$                     ; REPEAT
                                093D  1041
                  51    04    C4  093D  1042 20$:     MULL    #4,R1                   ; MULTIPLY # OF ENTRIES BY 4
                  51    0C    C0  0940  1043          ADDL    #12,R1                  ; ADD IN OVERHEAD
                     08D4    30  0943  1044          BSBW    ALONONPAGED             ; ALLOCATE MEMORY
              08 A2    51    B0  0946  1045          MOVW    R1,8(R2)                ; STORE SIZE OF STRUCTURE
              0A A2    38    90  094A  1046          MOVB    #DYN$C_RSHT,10(R2)      ; STORE STRUCTURE TYPE
    00000000'EF    0C A2    9E  094E  1047          MOVAB   12(R2),LCK$GL_HASHTBL   ; STORE POINTER TO HASH TABLE
       00000000'EF    54    D0  0956  1048          MOVL    R4,LCK$GL_HTBLCNT       ; STORE POWER OF TWO COUNT OF ENTRIES
    00000000'EF    54    10    83  095D  1049          SUBB3   #16,R4,LCK$GB_HTBLSHFT  ; STORE IT AS A RIGHT SHIFT COUNT
                                0965  1050                                          ; NOTE: HASH TABLE INITIALIZED TO ZERO
                                0965  1051 ;
                                0965  1052 ; ALLOCATE PROCESS BITMAP.  THIS BITMAP HAS ONE BIT FOR EVERY POSSIBLE
                                0965  1053 ; PROCESS IN THE SYSTEM.
                                0965  1054 ;
                                0965  1055 INI_PRCBITMAP:
    51   00000000'EF    3C    0965  1056          MOVZWL  SGN$GW_MAXPRCCT,R1      ; GET MAX. # OF PROCESSES IN SYSTEM
```

```
                51    08   C6  096C  1057          DIVL    #8,R1                       ; COMPUTE # OF BYTES NEEDED
                51    0D   C0  096F  1058          ADDL    #13,R1                      ; ADD IN OVERHEAD PLUS EXTRA BYTE
                                0972  1059                                              ; TO HANDLE TRUNCATION ERROR
                   08A5   30  0972  1060          BSBW    ALONONPAGED                 ; ALLOCATE MEMORY
           08 A2    51    B0  0975  1061          MOVW    R1,8(R2)                    ; STORE SIZE OF STRUCTURE
       0A A2   0563 8F    B0  0979  1062          MOVW    #<DYN$C_PRCMAP@8>+DYN$C_INIT,10(R2) ; STORE STRUCTURE TYPE
       04 A2    51    0C   C3  097F  1063          SUBL3   #12,R1,4(R2)                ; STORE SIZE OF BITMAP PORTION ONLY
00000000'EF     0C A2   9E  0984  1064          MOVAB   12(R2),LCK$GL_PRCMAP        ; STORE POINTER TO BITMAP
```

```
                                098C  1066          .SBTTL  Initialize Process State
                                098C  1067 ;
                                098C  1068 ; INIT PROCESS STATE
                                098C  1069 ;
                                098C  1070
                                098C  1071 INI_PSTATE:                                  ;
        51    00000000'EF       3C  098C  1072          MOVZWL  SGN$GW_MAXPRCCT,R1      ; GET TOTAL COUNT OF PROCESSES
                    51          D6  0993  1073          INCL    R1                      ; ADD ONE FOR SYSTEM HEADER
                  51    06      C4  0995  1074          MULL    #6,R1                   ; 2 BYTES FOR SEQ + 4 BYTES FOR PCB ADDR
                  51    0C      C0  0998  1075          ADDL    #DYN$C_HEADLEN,R1       ; ADD IN HEADER LENGTH
                     087C       30  099B  1076          BSBW    ALONONPAGED             ; ALLOCATE SPACE FOR SEQUENCE VECTOR
                                099E  1077                                              ; AND PCB VECTOR
                    82          7C  099E  1078          CLRQ    (R2)+                   ; CLEAR OUT TRASH
                  82    51      B0  09A0  1079          MOVW    R1,(R2)+                ; SET IN SIZE
              82    0163 8F     B0  09A3  1080          MOVW    #<DYN$C_PCBVEC@8!DYN$C_INIT>,(R2)+ ; SET TYPE AND SUBTYPE
        00000000'EF    52       D0  09A8  1081          MOVL    R2,SCH$GL_PCBVEC        ; SAVE POINTER TO PCB VECTOR
        51    00000000'EF       3C  09AF  1082          MOVZWL  SGN$GW_MAXPRCCT,R1      ; GET COUNT OF PROCESSES
    00000000'EF    04 A241      DE  09B6  1083          MOVAL   4(R2)[R1],SCH$GL_SEQVEC ; SET BASE OF SEQUENCE VECTOR
      6241    00000000'EF       DE  09BF  1084          MOVAL   MMG$AL_SYSPCB,(R2)[R1]  ; SET POINTER TO SYSTEM PCB
              50    6241        D0  09C7  1085          MOVL    (R2)[R1],R0             ; GET POINTER TO SYSTEM PCB
             60 A0    51       B0  09CB  1086          MOVW    R1,PCB$L_PID(R0)        ; SET PROPER PIX FOR SYSTEM PCB
                    51          D7  09CF  1087          DECL    R1                      ; COMPUTE MAXIMUM PIX VALUE
        00000000'EF    51       D0  09D1  1088          MOVL    R1,SCH$GL_MAXPIX        ; AND SET
                    50    1F    D0  09D8  1089          MOVL    #31,R0                  ; START AT HIGH BIT
             03 51    50       E0  09DB  1090 10$:      BBS     R0,R1,11$               ; FIND THE HIGHEST SET BIT
                 F9 50          F5  09DF  1091          SOBGTR  R0,10$                  ; WE KNOW WE WILL FIND ONE
    00000000'EF    50    01      C1  09E2  1092 11$:      ADDL3   #1,R0,SCH$GL_PIXWIDTH  ; BIT "N" SET MEANS THAT PIX WIDTH FOR
                                09EA  1093                                              ;  EXTENDED PID IS "N+1" BITS
      6241    00000000'EF       DE  09EA  1094 20$:      MOVAL   SCH$GL_NULLPCB,(R2)[R1] ; INIT VECTOR TO POINT TO NULL PROCESS
          00000000'FF41         B4  09F2  1095          CLRW    @SCH$GL_SEQVEC[R1]      ; INITIALIZE SEQUENCE COUNTER
                EE 51           F4  09F9  1096          SOBGEQ  R1,20$                  ; INIT THEM ALL
    00000000'EF    00000000'EF    D0  09FC  1097          MOVL    MMG$GL_GPTE,EXE$GL_GPT  ; ESTABLISH POINTER TO GLOBAL FREE LIST
                    52          D4  0A07  1098          CLRL    R2                      ; SET NULL PRIO INCR CLASS
          54    00000000'EF       DE  0A09  1099          MOVAL   SCH$GL_NULLPCB,R4       ; GET ADDRESS OF NULL PCB
                    0B          10  0A10  1100          BSBB    30$                     ; SETUP PROCESS PHYPCB AND STATE
          54    00000000'EF       DE  0A12  1101          MOVAL   SCH$GL_SWPPCB,R4        ; GET ADDRESS OF SWAPPER PCB
                    02          10  0A19  1102          BSBB    30$                     ; SETUP PROCESS PHYPCB AND STATE
                    36          11  0A1B  1103          BRB     INI_PHV                 ; CONTINUE WITH INITIALIZATION
  50    18 A4    15    09       EF  0A1D  1104 30$:      EXTZV   #VA$V_VPN,#VA$S_VPN,PCB$L_PHYPCB(R4),R0 ; GET VPN FOR HW PCB
    50    00000000'FF40         D0  0A23  1105          MOVL    @MMG$GL_SPTBASE[R0],R0  ; TRANSLATE TO ACTUAL PHYSICAL
  18 A4    15    09    50       F0  0A2B  1106          INSV    R0,#VA$V_VPN,#PTE$S_PFN,PCB$L_PHYPCB(R4); SET PHYSICAL ADDRESS
              50    60 A4       D0  0A31  1107          MOVL    PCB$L_PID(R4),R0        ; GET INTERNAL PROCESS IDENTIFIER
                 51    50       D0  0A35  1108          MOVL    R0,R1                   ; AND SAVE A COPY
              00000000'EF       16  0A38  1109          JSB     EXE$IPID_TO_EPID        ; CONVERT INTERNAL TO EXTENDED PROCESS ID
               64 A4    50       D0  0A3E  1110          MOVL    R0,PCB$L_EPID(R4)       ; STORE THE EXTENDED PROCESS ID
                 50    51       3C  0A42  1111          MOVZWL  R1,R0                   ; GET PROCESS INDEX FROM INTERNAL PID
      00000000'FF40    54       D0  0A45  1112          MOVL    R4,@SCH$GL_PCBVEC[R0]   ; AND SET ADDRESS IN PCB VECTOR
              00000000'EF       17  0A4D  1113          JMP     SCH$CHSE                ; AND CHANGE STATE TO EXECUTABLE
                                0A53  1114
                                0A53  1115 ;
                                0A53  1116 ;       INITIALIZE PROCESS HEADER VECTOR
                                0A53  1117 ;
                                0A53  1118 INI_PHV:                                     ;
  54    00000000'EF    01       C1  0A53  1119          ADDL3   #1,SGN$GL_BALSETCT,R4   ; GET COUNT OF SLOTS
              51    54    02    78  0A5B  1120          ASHL    #2,R4,R1                ; SIZE OF BLOCK TO ALLOCATE
                    51    0C    C0  0A5F  1121          ADDL    #DYN$C_HEADLEN,R1       ; ADD IN SIZE OF HEADER
                       07B5     30  0A62  1122          BSBW    ALONONPAGED             ; ALLOCATE BLOCK FOR PHV
```

```
                    82   7C   0A65  1123          CLRQ    (R2)+                           ; CLEAR OUT TRASH
                82  51   B0   0A67  1124          MOVW    R1,(R2)+                        ; SET IN SIZE
            82  0263 8F   B0   0A6A  1125          MOVW    #<DYN$C_PHVEC@8!DYN$C_INIT>,(R2)+ ; SET TYPE AND SUBTYPE
        00000000'EF  52   D0   0A6F  1126          MOVL    R2,PHV$GL_PIXBAS                ; SET PROCESS INDEX VECTOR
    00000000'EF  6244 3E   0A76  1127          MOVAW   (R2)[R4],PHV$GL_REFCBAS ; AND REFERENCE COUNT BASE
FE A244  00000000'EF  B0   0A7E  1128          MOVW    SGN$GW_MAXPRCCT,-2(R2)[R4]; SET SYSTEM PIX
                    54   D7   0A87  1129          DECL    R4                              ;
00000000'FF44  0400 8F   B0   0A89  1130          MOVW    #1024,@PHV$GL_REFCBAS[R4]       ; SET SYSTEM REFERENCE COUNT
                    54   D7   0A93  1131 20$:     DECL    R4                              ;
                    0D   19   0A95  1132          BLSS    30$                             ; CHECK FOR DONE
    00000000'FF44   01   AE   0A97  1133          MNEGW   #1,@PHV$GL_REFCBAS[R4]  ; INIT REFC
                6244 B4   0A9F  1134          CLRW    (R2)[R4]                        ; AND PIX
                    EF   11   0AA2  1135          BRB     20$                             ; AND AGAIN
                         0AA4  1136 30$:                                                ;
                         0AA4  1137 ;
                         0AA4  1138 ;       SETUP SWAPPER MAP
                         0AA4  1139 ;
                         0AA4  1140 INI_SWAP:                                           ;
51  00000000'EF  02   78   0AA4  1141          ASHL    #2,SGN$GL_MAXWSCNT,R1   ; GET SIZE TO ALLOCATE
                51  10   C0   0AAC  1142          ADDL    #4+DYN$C_HEADLEN,R1     ; ADD SPACE FOR STOPPER LONGWORD + HEAD
                    0768 30   0AAF  1143          BSBW    ALONONPAGED                     ; ALLOCATE A BLOCK FOR SWAPPER MAP
                    82   7C   0AB2  1144          CLRQ    (R2)+                           ; CLEAR OUT TRASH
                82  51   B0   0AB4  1145          MOVW    R1,(R2)+                        ; SET IN SIZE
            82  0363 8F   B0   0AB7  1146          MOVW    #<DYN$C_SWPMAP@8!DYN$C_INIT>,(R2)+ ; SET TYPE AND SUBTYPE
        00000000'EF  52   D0   0ABC  1147          MOVL    R2,SWP$GL_MAP                   ; SET ADDRESS OF SWAPPER MAP
        54  00000000'EF   DE   0AC3  1148          MOVAL   SCH$GL_SWPPCB,R4                ; GET ADDRESS OF SWAPPER PCB
            55  6C A4   D0   0ACA  1149          MOVL    PCB$L_PHD(R4),R5                ; AND GET HEADER ADDRESS
        00C8 C5  52   D0   0ACE  1150          MOVL    R2,PHD$L_POBR(R5)               ; SET BASE REGISTER
18  00  00000000'EF   F0   0AD3  1151          INSV    SGN$GL_MAXWSCNT,#0,#24,PHD$L_POLRASTL(R5) ; AND LENGTH REGISTER
                00CC C5          0ADB
                         0ADE  1152 ;
                         0ADE  1153 ; ALLOCATE MODIFIED PAGE WRITER PAGE TABLE ENTRY ARRAY
                         0ADE  1154 ; AND PROCESS HEADER VECTOR INDEX ARRAY.
                         0ADE  1155 ;
                         0ADE  1156 INI_MPW:
    51  00000000'EF   3C   0ADE  1157          MOVZWL  MPW$GW_MPWPFC,R1        ; MODIFIED PAGE WRITER PAGE FAULT CLUSTER
            51  07   CA   0AE5  1158          BICL    #7,R1                           ; TRUNCATE TO MULTIPLE OF 8
                03   12   0AE8  1159          BNEQ    10$                             ; CANNOT ALLOW ZERO
            51  10   D0   0AEA  1160          MOVL    #16,R1                          ; USE MINIMUM INSTEAD
    00000000'EF  51   B0   0AED  1161 10$:     MOVW    R1,MPW$GW_MPWPFC                ; RESET PARAMETER
    00000000'EF  51   B0   0AF4  1162          movw    r1,swp$gw_swpinc                ; ********** temp **********
            51  07   C0   0AFB  1163          ADDL2   #7,R1                           ; ALLOW BIT-LEVEL PAGE FILE ALLOCATION
        54  51  02   9C   0AFE  1164          ROTL    #2,R1,R4                        ; SIZE OF PTE ARRAY
        51  0C A441   3E   0B02  1165          MOVAW   DYN$C_HEADLEN(R4)[R1],R1; 6 BYTES PER PAGE TO ALLOCATE + HEADER
                0710 30   0B07  1166          BSBW    ALONONPAGED                     ; ALLOCATE THE STORAGE
                    82   7C   0B0A  1167          CLRQ    (R2)+                           ; CLEAR OUT TRASH
                82  51   B0   0B0C  1168          MOVW    R1,(R2)+                        ; SET IN SIZE
            82  0463 8F   B0   0B0F  1169          MOVW    #<DYN$C_MPWMAP@8!DYN$C_INIT>,(R2)+ ; SET TYPE AND SUBTYPE
        00000000'EF  52   D0   0B14  1170          MOVL    R2,MPW$AL_PTE                   ; ADR OF PAGE TABLE ENTRY ARRAY
00000000'EF  54  52   C1   0B1B  1171          ADDL3   R2,R4,MPW$AW_PHVINDEX   ; ADR OF PROCESS HEADER VECTOR INDEX ARRAY
```

```
                                0B23  1173          .SUBTITLE       MISCELLANEOUS INITIALIZATION
                                0B23  1174 ;
                                0B23  1175 ; DO SHORT PIECES OF MISCELLANEOUS SYSTEM INITIALIZATION
                                0B23  1176 ;
                                0B23  1177 INI_MISC:
                                0B23  1178 ;
                                0B23  1179 ; INITIALIZE THE GLOBAL PAGE FILE LIMIT
                                0B23  1180 ;
     00000000'EF    00000000'EF  D0  0B23  1181          MOVL    SGN$GL_GBLPAGFIL,MMG$GL_GBLPAGFIL
                                0B2E  1182 ;
                                0B2E  1183 ; INITIALIZE PAGEFILE CONTROL BLOCK 0 FOR READ OF SHELL INTO SYSTEM WORKING SET
                                0B2E  1184 ;
                 00000002'8F    D0  0B2E  1185          MOVL    #<<SWP$GL_SHELLBAS&^X7FFFFFFF>a-9>+1+1,- ;SET STARTING VBN
                 00000010'EF        0B34  1186                  MMG$GL_NULPFL+PFL$L_VBN         ; FOR SHELL PROCESS
                50   0000'CF    D0  0B39  1187          MOVL    W^BOO$GL_BOOTCB,R0              ; GET BOOT CONTROL BLOCK ADDR
                       14 A0    D0  0B3E  1188          MOVL    BOO$L_SYS_MAP(R0),-             ; STORE ADDRESS OF SYSTEM WCB
                 0000000C'EF        0B41  1189                  MMG$GL_NULPFL+PFL$L_WINDOW      ; FOR SHELL READS INTO SYSTEM
00000000'EF    00000000'EF  01  A1  0B46  1190          ADDW3   #1,SGN$GW_SWPFILES,SGN$GW_SWPFILCT ; SET ACTUAL NUMBER OF SWAP
                                0B52  1191                                                  ; FILE SLOTS
                                0B52  1192 ;
                                0B52  1193 ; GIVE RPB A PFN DATA BSE
                                0B52  1194 ;
   51   00000000'EF    16  09  EF  0B52  1195          EXTZV   #9,#22,EXE$GL_RPB,R1            ; GET VIRTUAL PAGE NUMBER
  00000000'FF41    FFE00000 8F  CB  0B58  1196          BICL3   #^C<PTE$M_PFN>,@MMG$GL_SPTBASE[R1],R0 ; GET PFN
                         50      0B67
 00000000'FF40    00000000'FF41  DE  0B68  1197          MOVAL   @MMG$GL_SPTBASE[R1],@PFN$AL_PTE[R0] ; PTE BACK POINTER
                 00000000'FF40  B6  0B75  1198          INCW    @PFN$AW_REFCNT[R0]              ; REF COUNT
            00000000'FF40   07  90  0B7C  1199          MOVB    #PFN$C_ACTIVE,@PFN$AB_STATE[R0] ; STATE IS ACTIVE
            00000000'FF40   01  90  0B84  1200          MOVB    #1,@PFN$AB_TYPE[R0]             ; SYSTEM PAGE
                                0B8C  1201
```

```
                     0B8C  1203          .SUBTITLE       PAGE AND SWAP FILE VECTOR INITIALIZATION
                     0B8C  1204 ;--
                     0B8C  1205 ;
                     0B8C  1206 ;  Functional Description:
                     0B8C  1207 ;
                     0B8C  1208 ;       The page file control block vector is initialized. This vector
                     0B8C  1209 ;       contains a longword pointer for each page file or swap file recognized
                     0B8C  1210 ;       by the system. Each vector element is initialized to point to a
                     0B8C  1211 ;       dummy PFL allocated in SYS.EXE. As a new page file or swap file is
                     0B8C  1212 ;       added to the system, a vector slot is loaded with its PFL address.
                     0B8C  1213 ;
                     0B8C  1214 ;  Input Parameters:
                     0B8C  1215 ;
                     0B8C  1216 ;       SGN$GW_PAGFILCT          Maximum number of paging files
                     0B8C  1217 ;       SGN$GW_SWPFILCT          Maximum number of swapping files
                     0B8C  1218 ;
                     0B8C  1219 ;  Implicit Input:
                     0B8C  1220 ;
                     0B8C  1221 ;       none
                     0B8C  1222 ;
                     0B8C  1223 ;  Output Parameters:
                     0B8C  1224 ;
                     0B8C  1225 ;       None
                     0B8C  1226 ;
                     0B8C  1227 ;  Implicit Output:
                     0B8C  1228 ;
                     0B8C  1229 ;       An array of longwords is allocated from nonpaged pool to contain
                     0B8C  1230 ;       the page file control block vector.
                     0B8C  1231 ;
                     0B8C  1232 ;  Completion Status:
                     0B8C  1233 ;
                     0B8C  1234 ;       If allocation of the vector of longwords fails, the bootstrap
                     0B8C  1235 ;       operation is aborted.
                     0B8C  1236 ;
                     0B8C  1237 ;--
                     0B8C  1238
                     0B8C  1239 INI_PFLVEC:
50   00000000'GF  3C  0B8C  1240          MOVZWL  G^SGN$GW_PAGFILCT,R0     ; Zero extend page file count
51   00000000'GF  3C  0B93  1241          MOVZWL  G^SGN$GW_SWPFILCT,R1     ;  ... and swap file count
     52   51   50  C1  0B9A  1242          ADDL3   R0,R1,R2                 ; Form their sum
               52  DD  0B9E  1243          PUSHL   R2                       ;  ... and save this result
     51   52   04  C5  0BA0  1244          MULL3   #4,R2,R1                 ; Make R1 a byte count
          51   10  C0  0BA4  1245          ADDL2   #PTR$K_LENGTH,R1         ; Add header overhead
               51  DD  0BA7  1246          PUSHL   R1                       ; Save requested size
             066E  30  0BA9  1247          BSBW    ALONONPAGED              ; Allocate the pointer block
       08 A2   8E  D0  0BAC  1248          MOVL    (SP)+,PTR$W_SIZE(R2)     ; Size is the requested size
       0A A2   25  90  0BB0  1249          MOVB    #DYN$C_PTR,PTR$B_TYPE(R2) ; Set structure type as PTR
       0B A2   23  90  0BB4  1250          MOVB    #DYN$C_PFL,PTR$B_PTRTYPE(R2) ; ... which locates PFLs
       0C A2   6E  D0  0BB8  1251          MOVL    (SP),PTR$L_PTRCNT(R2)    ; Store size of PFL array
53   00000000'GF  DE  0BBC  1252          MOVAL   G^MMG$GL_NULLPFL,R3      ; Set up contents of uninitialized slot
       51   10 A2  DE  0BC3  1253          MOVAL   PTR$L_PTR0(R2),R1        ; Get address of first slot
00000000'GF     51  D0  0BC7  1254          MOVL    R1,G^MMG$GL_PAGSWPVC     ; Store this for exec routines
                50 8ED0  0BCE  1255          POPL    R0                       ; R0 is loop counter
          81    53  D0  0BD1  1256 10$:      MOVL    R3,(R1)+                 ; Initialize next slot
            FA  50  F5  0BD4  1257          SOBGTR  R0,10$                   ; If not done, load next slot
```

```
                                0BD7  1259 ;
                                0BD7  1260 ;       INITIALIZE POINTER TO TOP OF INTERRUPT STACK AND COMPUTE MAXIMUM
                                0BD7  1261 ;       ALLOWED DEPTH FOR LOCK MANAGER RESOURCE NAMES
                                0BD7  1262 ;
                                0BD7  1263 INI_INTSTKLM:
      50   00000000'EF   3C     0BD7  1264         MOVZWL  SGN$GW_ISPPGCT,R0       ; GET # OF PAGES OF INTERRUPT STACK
         50   50   09   78     0BDE  1265         ASHL    #9,R0,R0                ; CONVERT TO BYTES
00000000'EF   50   C3     0BE2  1266         SUBL3   R0,EXE$GL_INTSTK,-      ; SUBTRACT FROM BASE OF STACK AND STORE
         00000000'EF             0BE9  1267                 EXE$GL_INTSTKLM
      50   00000000'EF   C2     0BEE  1268         SUBL    LCK$GL_EXTRASTK,R0      ; SUBTRACT EXTRA STACK AMOUNT FROM
                                0BF5  1269                                 ; SIZE OF INTERRUPT STACK
            50   10   C6     0BF5  1270         DIVL    #16,R0                  ; ALLOW 16 BYTES FOR EACH LEVEL
            04   50   D1     0BF8  1271         CMPL    R0,#4                   ; MAKE SURE IT'S AT LEAST 4
                      03   18     0BFB  1272         BGEQ    10$                     ; IT IS
            50   04   D0     0BFD  1273         MOVL    #4,R0                   ; SET IT TO 4
000000FF   8F   50   D1     0C00  1274 10$:    CMPL    R0,#255                 ; MAKE SURE IT'S NO MORE THAN 255
                      07   15     0C07  1275         BLEQ    20$                     ; IT IS
      50   000000FF   8F   D0     0C09  1276         MOVL    #255,R0                 ; SET IT TO 255
00000000'EF   50   90     0C10  1277 20$:    MOVB    R0,LCK$GB_MAXDEPTH      ; STORE IT
                                0C17  1278 ;
                                0C17  1279 ; POINT SYSTEM PCB AT SYSTEM PHD AND SET PROPER PIX
                                0C17  1280 ;
                                0C17  1281 INI_SYSPCB:                             ;
   54   00000000'EF   DE     0C17  1282         MOVAL   MMG$AL_SYSPCB,R4        ; GET ADDRESS OF SYSTEM PCB
6C A4   00000000'EF   D0     0C1E  1283         MOVL    MMG$GL_SYSPHD,PCB$L_PHD(R4)     ; AND SET IN SYSTEM PCB
60 A4   00000000'EF   B0     0C26  1284         MOVW    SGN$GW_MAXPRCCT,PCB$L_PID(R4)   ; SET SYSTEM PIX
                                0C2E  1285 ;
                                0C2E  1286 ; SYSBOOT PRODUCED A MAP FOR SYS.EXE AND LEFT IT IN THE BOOT CONTROL
                                0C2E  1287 ; BLOCK.  ITS RETRIEVAL POINTERS ARE IN THE NORMALIZED FORM OF 32 BITS
                                0C2E  1288 ; OF BLOCK COUNT, 32 BITS OF LBN.  SYSBOOT LEFT ENOUGH SPACE AT THE END
                                0C2E  1289 ; OF THE MAP TO MAKE A REAL WCB AND PUT IT RIGHT ON TOP OF THIS MAP.
                                0C2E  1290 ;
   58   0000'CF   D0     0C2E  1291         MOVL    W^BOO$GL_BOOTCB,R8      ; ADDRESS OF BOOT CONTROL BLOCK
      56   14   A8   D0     0C33  1292         MOVL    BOO$L_SYS_MAP(R8),R6    ; ADDRESS OF MAP LEFT BY SYSBOOT
57   66   FD   8F   78     0C37  1293         ASHL    #-3,(R6),R7             ; RETRIEVAL POINTER COUNT IN MAP
   F461   CF   57   B0     0C3C  1294         MOVW    R7,W^SYSWCB+WCB$W_NMAP  ; SET COUNT IN TEMPLATE WCB
      50   04   A6   DE     0C41  1295         MOVAL   4(R6),R0                ; ADR OF 1ST 8 BYTE RTRV PTR
         51   56   D0     0C45  1296         MOVL    R6,R1                   ; ADR TO STORE 1ST 6 BYTE RTRV PTR
         52   57   D0     0C48  1297         MOVL    R7,R2                   ; NUMBER OF RETRIEVAL POINTERS
                                0C4B  1298 ;
                                0C4B  1299 ; COLLAPSE THE 8 BYTE FORMAT INTO A 6 BYTE FORMAT.  SINCE SYS.EXE ITSELF
                                0C4B  1300 ; IS NOT 65K BLOCKS BIG, NONE OF THESE RETRIEVAL POINTERS CAN HAVE A
                                0C4B  1301 ; NON-ZERO HIGH ORDER WORD.
                                0C4B  1302 ;
         81   80   F7     0C4B  1303 20$:    CVTLW   (R0)+,(R1)+             ; MOVE THE BLOCK COUNT
         81   80   D0     0C4E  1304         MOVL    (R0)+,(R1)+             ; AND THE STARTING LBN
            F7   52   F5     0C51  1305         SOBGTR  R2,20$                  ; LOOP THROUGH ALL RTRV PTRS
         57   06   C4     0C54  1306         MULL    #6,R7                   ; NUMBER OF BYTES IN NEW RTRV PTRS
   F438   CF   57   A0     0C57  1307         ADDW    R7,W^SYSWCB+WCB$W_SIZE  ; FIX UP TEMPLATE WCB SIZE FIELD
30 A6   66   57   28     0C5C  1308         MOVC3   R7,(R6),WCB$C_LENGTH(R6) ; MOVE 6 BYTE RTRV PTRS DOWN
66   F426   CF   30   28     0C61  1309         MOVC3   #WCB$C_LENGTH,W^SYSWCB,(R6) ; AND INSERT THE TEMPLATE WCB
                                0C67  1310 ;
                                0C67  1311 ; Compute the data that calibrates the time-wait loop, used by drivers
                                0C67  1312 ; to wait, instead of reading the processor clock.
                                0C67  1313 ;
                                0C67  1314
   00000000'EF   16     0C67  1315         JSB     EXE$INI_TIMWAIT         ; INITIALIZE DATA FOR TIMEWAIT LOOP
```

```
                                0C6D  1316 ;
                                0C6D  1317 ; INSERT ALL DRIVERS LINKED AS PART OF THE SYSTEM IMAGE IN THE DRIVER
                                0C6D  1318 ; PROLOGUE TABLE LIST SO THAT SUBSEQUENT DEVICE CONNECTIONS WILL BE ABLE
                                0C6D  1319 ; TO FIND THEM.
                                0C6D  1320 ;
00000000'EF   00000000'EF   0E  0C6D  1321          INSQUE  MB$DPT,IOC$GL_DPTLIST   ; INSERT MAILBOX DRIVER ON DRIVER LIST
00000000'EF   00000000'EF   0E  0C78  1322          INSQUE  NL$DPT,IOC$GL_DPTLIST   ; INSERT NULL DRIVER ON DRIVER LIST
00000000'EF   00000000'EF   0E  0C83  1323          INSQUE  OP$DPT,IOC$GL_DPTLIST   ; INSERT CONSOLE DRIVER ON DRIVER LIST
                                0C8E  1324 ;
                                0C8E  1325 ; move logical name for SYS$DISK and SYS$SYSDEVICE into non-paged pool
                                0C8E  1326 ;
                                0C8E  1327 INI_LOG:                                 ;
          51   0074 8F   3C     0C8E  1328          MOVZWL  #BDL$S_CRELNM_ITMLST,R1 ; get size to allocate
                     0584   30   0C93  1329          BSBW    ALONONPAGED             ; allocate it
     00000000'FF    52   D0     0C96  1330          MOVL    R2,@LNM$AL_HASHTBL      ; store addr temporarily (see swapinit)
          54   F35F CF   9E     0C9D  1331          MOVAB   BDL$GL_DISK_LOG,R4      ; get addr of block
               56   52   D0     0CA2  1332          MOVL    R2,R6                   ; save pointer
       62   64   0074 8F   28   0CA5  1333          MOVC3   #BDL$S_CRELNM_ITMLST,(R4),(R2)
                                0CAB  1334                                          ; copy block to pool
                                0CAB  1335          ;
                                0CAB  1336          ; make self relative pointer absolute
                                0CAB  1337          ;
            10 A6   56   C0     0CAB  1338          ADDL    R6,BDL_L_DISK_EQ_PTR(R6)
            04 A6   56   C0     0CAF  1339          ADDL    R6,BDL_L_DISK_AT_PTR(R6)
            44 A6   56   C0     0CB3  1340          ADDL    R6,BDL_L_SYSD_EQ_PTR(R6)
            38 A6   56   C0     0CB7  1341          ADDL    R6,BDL_L_SYSD_AT_PTR(R6)
                                0CBB  1342 ;
                                0CBB  1343 ; THE TERMINAL SERVICE IS NOW ALLOCATED NON-PAGED POOL AND READ INTO IT
                                0CBB  1344 ; BY SYSBOOT.  THE ADDRESS IN POOL IS PASSED IN BOO$GL_TRMDRV.
                                0CBB  1345 ; INITIALIZE THE TERMINAL DRIVER.
                                0CBB  1346 ;
                                0CBB  1347 INI_TTYDRV:                              ;
                                0CBB  1348          .ENABLE LSB
                                0CBB  1349 ;
                                0CBB  1350 ; INITIALIZE PARTS OF OPA0 UCB THAT CAN ONLY BE DONE AFTER SYSBOOT HAS
                                0CBB  1351 ; POSSIBLY CHANGED SYSGEN PARAMS
                                0CBB  1352 ;
54   00000000'GF    9E          0CBB  1353          MOVAB   G^OP$DPT,R4             ; R4 POINTS TO CONSOLE DPT
55   00000000'GF    9E          0CC2  1354          MOVAB   G^OPA$UCB0,R5           ; R5 POINTS TO CONSOLE UCB
     00000000'GF    16          0CC9  1355          JSB     G^IOC$INITDRV           ; INITIALIZE FIELDS THAT CAN'T BE DONE
                                0CCF  1356                                          ;  STATICALLY.
 00 00C4 C5     0C   E5         0CCF  1357          BBCC    #TT$V_SCOPE,W^UCB$L_TT_DECHAR(R5),10$ ; ENSURE NOT SCOPE
    00C8 C5     02   CA         0CD5  1358 10$:     BICL2   #TT2$M_AUTOBAUD,W^UCB$L_TT_DECHA1(R5) ; ENSURE NOT AUTOBAUD
    52   0000'CF   D0          0CDA  1359          MOVL    W^BOO$GL_TRMDRV,R2      ; ADDRESS OF TERMINAL SERVICE CODE
00000000'GF   52   D0          0CDF  1360          MOVL    R2,G^TTY$GL_DPT         ; STORE LOADED TERMINAL SERVICE DPT
00000000'GF   62   0E          0CE6  1361          INSQUE  (R2),G^IOC$GL_DPTLIST   ; INSERT TERMINAL DRIVER ON LIST
      0A A2   1E   91          0CED  1362          CMPB    #DYN$C_DPT,DPT$B_TYPE(R2) ; REALLY A DPT?
              0E   12          0CF1  1363          BNEQ    50$                     ; NOPE, GO SAY SO
           08 A5   B1          0CF3  1364          CMPW    UCB$W_SIZE(R5),-        ; CORRECT UCB SIZE IN CONSOLE UCB?
           0E A2                0CF6  1365                  DPT$W_UCBSIZE(R2)
              15   1E          0CF8  1366          BGEQU   70$                     ; IF GEQU YES
    51   F47C CF   9E          0CFA  1367          MOVAB   BADCONUCB,R1            ; SET ILLEGAL OPA0: ERROR
              05   11          0CFF  1368          BRB     OUTZ                    ; AND GO DIE
                                0D01  1369
    51   F4C5 CF   9E          0D01  1370 50$:     MOVAB   BADTTYDRV,R1            ; SET ILLEGAL FORMAT ERROR
              5B   D4          0D06  1371 OUTZ:    CLRL    R11                     ; ADDRESS CONSOLE
     00000000'EF   16          0D08  1372          JSB     EXE$OUTZSTRING          ; OUTPUT IT
```

```
                 00  0D0E  1373          HALT                                ;
                     0D0F  1374 ;
                     0DCF  1375 ; RELOCATE TTDRIVER CLASS VECTOR, SET ADDRESS OF OPA0 DDT
                     0D0F  1376 ;
          51   1E A2   3C  0D0F  1377 70$:    MOVZWL  DPT$W_VECTOR(R2),R1         ; OFFSET TO CLASS VECTOR DISPATCH TABLE
             51   52   C0  0D13  1378         ADDL2   R2,R1                       ; CALCULATE ADDRESS
             53   51   D0  0D16  1379         MOVL    R1,R3                       ; SAVE
                       0D19  1380
             81   52   C0  0D19  1381 80$:    ADDL2   R2,(R1)+                    ; ADD IN DPT OFFSET
                  61   D5  0D1C  1382         TSTL    (R1)                        ; END OF LIST ?
                  F9   12  0D1E  1383         BNEQ    80$                         ; BRANCH IF NO
                       0D20  1384
               10 A3   D0  0D20  1385         MOVL    CLASS_DDT(R3),-
 0000000C'EF           0D23  1386                 OPA$GL_DDB+DDB$L_DDT            ; STORE DDT IN CONSOLE DDB
               10 A3   D0  0D28  1387         MOVL    CLASS_DDT(R3),-
             0088 C5       0D2B  1388                 UCB$L_DDT(R5)               ; STORE DDT IN CONSOLE UCB
          0114 C5   53   D0  0D2E  1389         MOVL    R3,-
                           0D33  1390                 UCB$L_TT_CLASS(R5)          ; STORE CLASS VECTOR ADDRESS
                  63   D0  0D33  1391         MOVL    CLASS_GETNXT(R3),-
             010C C5       0D35  1392                 UCB$L_TT_GETNXT(R5)         ; STORE GET NEXT IN CONSOLE UCB
               04 A3   D0  0D38  1393         MOVL    CLASS_PUTNXT(R3),-
             0110 C5       0D3B  1394                 UCB$L_TT_PUTNXT(R5)         ; STORE PUT NEXT IN CONSOLE UCB
          51   1E A4   3C  0D3E  1395         MOVZWL  DPT$W_VECTOR(R4),R1         ; OFFSET TO CONSOLE PORT VECTOR DISPATCH TAB
0118 C5   54   51   C1  0D42  1396         ADDL3   R1,R4,-
                           0D48  1397                 UCB$L_TT_PORT(R5)           ; SET ADDRESS IN CONSOLE UCB
                           0D48  1398         .DISABLE LSB
```

```
                        0D48  1400          .SBTTL  INIT THE BOOT DEVICE
                        0D48  1401 ;
                        0D48  1402 ; THE BOOTSTRAP DEVICE DRIVER'S ADDRESS IN POOL IS PASSED IN BOO$GL_DSKDRV.
                        0D48  1403 ; FINISH ALLOCATING AND INITIALIZING THE DATA BASE TO DESCRIBE THE BOOT DEVICE,
                        0D48  1404 ; AND ANY CLASS/PORT DRIVERS ASSOCIATED WITH IT.
                        0D48  1405 ;
                        0D48  1406 INI_BOOTDEVIC:                               ;
                        0D48  1407 ;
                        0D48  1408 ; FIGURE OUT THE DEVICE NAME FROM THE ADAPTER.
                        0D48  1409 ;
56   00000000'GF   D0   0D48  1410          MOVL    G^EXE$GL_RPB,R6              ; ADDRESS OF RPB
       7E   01   CE   0D4F  1411          MNEGL   #1,-(SP)                     ; INIT ADAPTER COUNTS
            6E   DD   0D52  1412          PUSHL   (SP)                         ; ALLOW FOR 8 ADAPTERS
       7E   6E   7D   0D54  1413          MOVQ    (SP),-(SP)                   ; ALLOW FOR 16 ADAPTERS
54   FFFFFFFC'EF   DE   0D57  1414          MOVAL   IOC$GL_ADPLIST-ADP$L_LINK,R4 ; START OF ADAPTER LIST
     54   04 A4   D0   0D5E  1415 10$:      MOVL    ADP$L_LINK(R4),R4            ; MOVE TO NEXT ADAPTER
            1C   13   0D62  1416          BEQL    20$                          ; BRANCH IF NONE
     51   0E A4   3C   0D64  1417          MOVZWL  ADP$W_ADPTYPE(R4),R1         ; GET ADAPTER TYPE CODE
          6E41   96   0D68  1418          INCB    (SP)[R1]                     ; BUMP APPROPRIATE COUNT
  0B A4   6E41   90   0D6B  1419          MOVB    (SP)[R1],ADP$B_NUMBER(R4)    ; SET ADAPTER NUMBER
20 A6   0C A4   B1   0D70  1420          CMPW    ADP$W_TR(R4),RPB$L_BOOTR1(R6)    ; IS THIS THE BOOT ADAPTER?
            E7   12   0D75  1421          BNEQ    10$                          ; NO, KEEP LOOKING
     55   6E41   9A   0D77  1422          MOVZBL  (SP)[R1],R5                  ; YES, SAVE THE COUNT
        57   54   D0   0D7B  1423          MOVL    R4,R7                        ; AND ADP ADDRESS
             DE   11   0D7E  1424          BRB     10$                          ; AND CONTINUE THROUGH ALL ADPS
        5E   10   C0   0D80  1425 20$:     ADDL    #16,SP                       ; CLEAN STACK
                        0D83  1426 ;
                        0D83  1427 ; IF A BOOT DEVICE CONTROLLER LETTER WAS PASSED TO US IN AN INPUT REGISTER
                        0D83  1428 ; TO VMB, USE THAT AS THE CONTROLLER LETTER.
                        0D83  1429 ;
     0108 C6   95   0D83  1430          TSTB    RPB$B_CTRLLTR(R6)            ; WAS A CONTROLLER LETTER SPECIFIED?
            07   13   0D87  1431          BEQL    22$                          ; BRANCH IF NOT
55   0108 C6   9A   0D89  1432          MOVZBL  RPB$B_CTRLLTR(R6),R5         ; GET THE CONTROLLER LETTER
            55   D7   0D8E  1433          DECL    R5                           ; BECAUSE A=1 (0 = UNSPECIFIED)
  58   64 A6   3C   0D90  1434 22$:      MOVZWL  RPB$W_UNIT(R6),R8            ; PICK UP UNIT NUMBER FROM BOOT
                        0D94  1435 ;
                        0D94  1436 ; FIRST CHECK FOR ANY PORT DRIVER.  IF THERE IS ONE, THEN THE DDB AND UCB
                        0D94  1437 ; FOR IT MUST BE ALLOCATED, SINCE THEY ARE NOT BUILT IN.
                        0D94  1438 ;
54   0000'CF   D0   0D94  1439          MOVL    W^BOO$GL_PRTDRV,R4           ; ADR IN POOL OF BOOT PORT DRIVER
     59   54   D0   0D99  1440          MOVL    R4,R9                        ; SAVE AWAY
            03   12   0D9C  1441          BNEQ    25$                          ; THERE IS
          00CD   31   0D9E  1442          BRW     50$                          ; THERE IS NO PORT DRIVER
                        0DA1  1443
            64   0E   0DA1  1444 25$:     INSQUE  DPT$L_FLINK(R4),-
 00000000'GF             0DA3  1445                  G^IOC$GL_DPTLIST             ; INSERT DRIVER IN LIST
     51   44 8F   9A   0DA8  1446          MOVZBL  #DDB$K_LENGTH,R1             ; GET SIZE OF DDB
          046B   30   0DAC  1447          BSBW    ALONONPAGED                  ; GRAB SOME POOL
     53   52   D0   0DAF  1448          MOVL    R2,R3                        ; TRANSFER
  08 A3   51   B0   0DB2  1449          MOVW    R1,DDB$W_SIZE(R3)            ; SET THE SIZE
  0A A3   06   9B   0DB6  1450          MOVZBW  #DYN$C_DDB,DDB$B_TYPE(R3)    ; AND TYPE
51   00000000'GF   DE   0DBA  1451          MOVAL   G^IOC$GL_DEVLIST,R1          ; PICK UP SYSTEM DDB LIST POINTER
     52   61   D0   0DC1  1452 30$:      MOVL    DDB$L_LINK(R1),R2            ; CHASE DOWN THE LIST
            05   13   0DC4  1453          BEQL    40$                          ; UNITL THE END
     51   52   D0   0DC6  1454          MOVL    R2,R1                        ; TRANSFER POINTERS
            F6   11   0DC9  1455          BRB     30$                          ; AND CONTINUE
                        0DCB  1456
```

```
                61   53   D0  0DCB  1457 40$:    MOVL    R3,DDB$L_LINK(R1)          ; LINK US IN
                                0DCE  1458 ;       CLRL    DDB$L_LINK(R3)             ; AND SET AS END
                   025E   30    0DCE  1459         BSBW    FIX_DRV_NAME               ; FIX UP THE NAMES
          00000000'9F   DE    0DD1  1460         MOVAL   @#SCS$GA_LOCALSB,-         ; SET IN THE SYSTEM BLOCK BACK LINK
                34 A3         0DD7  1461                 DDB$L_SB(R3)
                   53   DD    0DD9  1462         PUSHL   R3                         ; SAVE POINTER TO DDB
            51   0E A4   3C    0DDB  1463         MOVZWL  DPT$W_UCBSIZE(R4),R1       ; PICK UP SIZE OF UCB
      51    00000058 8F   C0    0DDF  1464         ADDL2   #ORB$C_LENGTH,R1           ; ALLOCATE ORB ADJACENT TO THE UCB
                0431   30    0DE6  1465         BSBW    ALONONPAGED                ; GET IT
                   53 8ED0    0DE9  1466         POPL    R3                         ; RESTORE DDB
                   3E   BB    0DEC  1467         PUSHR   #^M<R1,R2,R3,R4,R5>        ; SAVE MOVC REGISTERS
62  51   00   6E   00   2C    0DEE  1468         MOVC5   #0,(SP),#0,R1,(R2)         ; CLEAR OUT BLOCK JUST ALLOCATED
                   3E   BA    0DF4  1469         POPR    #^M<R1,R2,R3,R4,R5>        ; RESTORE MOVC REGISTERS
            51   0E A4   3C    0DF6  1470         MOVZWL  DPT$W_UCBSIZE(R4),R1       ; RESTORE ORIGINAL UCB SIZE
            04 A3   52   D0    0DFA  1471         MOVL    R2,DDB$L_UCB(R3)           ; SET POINTER
            28 A2   53   D0    0DFE  1472         MOVL    R3,UCB$L_DDB(R2)           ; AND ANOTHER
         1C A2      52   51   C1    0E02  1473         ADDL3   R1,R2,UCB$L_ORB(R2)        ; SET ORB ADDRESS
                5A   52   D0    0E07  1474         MOVL    R2,R10                     ; COPY ADDRESS
                   82   7C    0E0A  1475         CLRQ    (R2)+                      ; STEP TO SIZE
                   82   51   B0    0E0C  1476         MOVW    R1,(R2)+                   ; SET SIZE
                   82   10   9B    0E0F  1477         MOVZBW  #DYN$C_UCB,(R2)+           ; SET TYPE
                   82   52   D0    0E12  1478         MOVL    R2,(R2)+                   ; SET ASTQFL
             62    0C AA   D0    0E15  1479         MOVL    UCB$L_ASTQFL(R10),(R2)     ; SET ASTQBL
                4C AA   DE    0E19  1480         MOVAL   UCB$L_IOQFL(R10),-
                4C AA         0E1C  1481                 UCB$L_IOQFL(R10)           ; SET IOQFL
                4C AA   DE    0E1E  1482         MOVAL   UCB$L_IOQFL(R10),-
                50 AA         0E21  1483                 UCB$L_IOQBL(R10)           ; AND IOQBL
             50    1C AA   D0    0E23  1484         MOVL    UCB$L_ORB(R10),R0          ; GET THE ORB ADDRESS
          0A A0    49 8F   90    0E27  1485         MOVB    #DYN$C_ORB,ORB$B_TYPE(R0)      ; SET BLOCK TYPE
       08 A0    0058 8F   B0    0E2C  1486         MOVW    #ORB$C_LENGTH,ORB$W_SIZE(R0)   ; SET BLOCK SIZE
          0B A0    01   90    0E32  1487         MOVB    #ORB$M_PROT_16,ORB$B_FLAGS(R0) ; SOGW PROTECTION WORD
                                0E36  1488
                                0E36  1489         ASSUME  ORB$L_ACL_DESC EQ ORB$L_ACL_COUNT+4
                                0E36  1490
                28 A0   7C    0E36  1491         CLRQ    ORB$L_ACL_COUNT(R0)        ; NO ACL AS YET
                                0E39  1492 ;
                                0E39  1493 ; NOTE: The following code depends on several assumptions:
                                0E39  1494 ;       1) We will get here only if there are both a port and class driver
                                0E39  1495 ;       2) All UNIBUS port drivers have the same bootdriver (BTD$K_UDA)
                                0E39  1496 ;       3) "REAL" physical unit numbers on HSC's are less than 12 bits.
                                0E39  1497 ;
                                0E39  1498 ;       The boot device name is built up of three parts:
                                0E39  1499 ;       1) The device mnemonic (e.g. DU,DM) is extracted from the driver
                                0E39  1500 ;       name for most devices, otherwise it is copied up from the boot
                                0E39  1501 ;       driver thru SYSBOOT.
                                0E39  1502 ;       2) The controller letter is the same as the port controller for
                                0E39  1503 ;       all UNIBUS ports.  For the HSC, it is always "A".  For the
                                0E39  1504 ;       emulated disks, it is derived from the unit number.
                                0E39  1505 ;       3) The unit number is extracted from the RPB.
                                0E39  1506 ;
000000D4'9F     58   B0    0E39  1507         MOVW    R8,@#SYS$GL_BOOTUCB+ -     ; STUFF THE MSCP UNIT NUMBER
                                0E40  1508                   UCB$W_MSCPUNIT           ;  WHERE IT BELONGS
                11   91    0E40  1509         CMPB    #BTD$K_UDA,-
             66 A6              0E42  1510                 RPB$B_DEVTYP(R6)           ; IF BOOTING FROM CI/HSC/EMULA
                1D   13    0E44  1511         BEQL    48$                        ;  (NO)
                0C   EF    0E46  1512         EXTZV   #MSCP$V_EU_DESIG,-         ; EXTRACT CONTROLLER DESIGNATOR
          55    58   03         0E48  1513                 #MSCP$S_EU_DESIG,R8,R5     ;  FROM UNIT NUMBER
```

```
                16  13  0E4B  1514          BEQL    48$                     ; NONE, SO LEAVE AT "DxA"
                55  D7  0E4D  1515          DECL    R5                      ; BECAUSE 1=A
                00  EF  0E4F  1516          EXTZV   #MSCP$V_EU_SUBU,-       ; PICK UP 3-BIT UNIT NUMBER
      51    58    03       0E51  1517                  #MSCP$S_EU_SUBU,R8,R1
                08  ED  0E54  1518          CMPZV   #MSCP$V_EU_CTYPE,-      ; IF CONTROLLER TYPE IS NOT OLD
      00    58    04       0E56  1519                  #MSCP$S_EU_CTYPE,R8,-
                           0E59  1520                  #MSCP$K_EMD_OLD
                05  13  0E59  1521          BEQL    45$                     ;  THEN
                00  EF  0E5B  1522          EXTZV   #MSCP$V_EU_NO,-         ; PICK UP 8-BIT UNIT NUMBER
      51    58    08       0E5D  1523                  #MSCP$S_EU_NO,R8,R1
            58    51  D0  0E60  1524 45$:   MOVL    R1,R8                   ; SET UNIT NUMBER
     000000D4'9F      B5  0E63  1525 48$:   TSTW    @#SYS$GL_BOOTUCB+ -     ; CHECK FOR SHADOWING
                           0E69  1526                  UCB$W_MSCPUNIT
                03  18  0E69  1527          BGEQ    50$                     ; NOT SHADOWING
            55    12  D0  0E6B  1528          MOVL    #^X12,R5                ; SHADOWING: MAKE CONTROLLER "DxS"
                           0E6E  1529 ;
                           0E6E  1530 ; NOW FIX UP BOOT DISK DEVICE
                           0E6E  1531 ;
      54    0000'CF   D0  0E6E  1532 50$:   MOVL    W^BOO$GL_DSKDRV,R4      ; ADR IN POOL OF BOOT DISK DRIVER
                64  0E  0E73  1533          INSQUE  DPT$L_FLINK(R4),-
       00000000'GF       0E75  1534                  G^IOC$GL_DPTLIST        ; INSERT DRIVER IN LIST
   53   00000000'GF  DE  0E7A  1535          MOVAL   G^SYS$GL_BOOTDDB,R3     ; GET DDB ADDRESS FOR BOOT DEVICE
              01AB  30  0E81  1536          BSBW    FIX_DRV_NAME            ; FIX UP THE NAMES
      51    0000'CF   B0  0E84  1537          MOVW    W^BOO$GL_DEVNAME,R1     ; PICK UP POSSIBLE BOOT DEVICE NAME
                04  13  0E89  1538          BEQL    55$                     ; THERE IS NONE
        15  A3    51  B0  0E8B  1539          MOVW    R1,DDB$T_NAME+1(R3)     ; STUFF IT IN DDB
       00000000'EF      D0  0E8F  1540 55$:   MOVL    CLU$GL_ALLOCLS,-        ; STORE DEVICE ALLOCATION CLASS
              3C  A3       0E95  1541                  DDB$L_ALLOCLS(R3)       ; IN DDB
         55   04  A3  D0  0E97  1542          MOVL    DDB$L_UCB(R3),R5        ; PICK UP UCB ADDRESS
              0E  A4  B1  0E9B  1543          CMPW    DPT$W_UCBSIZE(R4),-     ; CHECK THAT PREBUILT IS LARGE
              08  A5       0E9E  1544                  UCB$W_SIZE(R5)          ;  ENOUGH
                08  19  0EA0  1545          BLSS    60$                     ; YES
      51    F2FB  CF  9E  0EA2  1546          MOVAB   BADDSKUCB,R1            ; NO, SET ERROR MESSAGE
                FE5C  31  0EA7  1547          BRW     OUT2
                           0EAA  1548
         54  A5    58  B0  0EAA  1549 60$:   MOVW    R8,UCB$W_UNIT(R5)       ; SET PROPER UNIT NUMBER
00000000'GF    55  D0  0EAE  1550          MOVL    R5,G^EXE$GL_SYSUCB      ; SET ADDRESS OF SYSTEM DEVICE UCB
                0A  BB  0EB5  1551          PUSHR   #^M<R1,R3>              ; SAVE REGISTERS
      51    0000'CF   D0  0EB7  1552          MOVL    W^BOO$GL_BOOTCB,R1      ; GET BOOT CONTROL BLOCK ADDRESS
        50    14  A1  D0  0EBC  1553          MOVL    BOO$L_SYS_MAP(R1),R0    ; GET WCB ADDRESS FOR SYS.EXE
        10  A0    55  D0  0EC0  1554          MOVL    R5,WCB$L_ORGUCB(R0)     ; SET UCB IN WINDOW CONTROL BLOCK
00000000'EF    61  D0  0EC4  1555          MOVL    BOO$L_CHECKSUM(R1),EXE$GQ_BOOTCB_D ; BYTE COUNT TO CHECKSUM
                           0EC8  1556                                          ; LEFT HERE BY SYSBOOT
       00000000'EF  16  0ECB  1557          JSB     EXE$BOOTCB_CHK          ; CALCULATE BOOTCB CHECKSUM IN R3
            61  53  D0  0ED1  1558          MOVL    R3,BOO$L_CHECKSUM(R1)   ; AND STORE IT
                0A  BA  0ED4  1559          POPR    #^M<R1,R3>              ; RESTORE REGISTERS
     07  0D  A4    01  E1  0ED6  1560          BBC     #DPT$V_SVP,DPT$B_FLAGS(R4),70$  ; IF DRIVER REQUESTS ...
              038C  30  0EDB  1561          BSBW    ALOSPT                  ; ALLOCATE SPT SLOT FOR UCB WINDOW
        74  A5    50  D0  0EDE  1562          MOVL    R0,UCB$L_SVPN(R5)       ; SAVE SYSTEM VPN IN UCB
                           0EE2  1563 ;
                           0EE2  1564 ; NOW THE BOOT DISK DRIVER (AND ANY ASSOCIATED PORT DRIVER) HAVE BEEN HOOKED
                           0EE2  1565 ; INTO THE DATA BASE. DDB'S AND UCB'S ARE PRESENT.
                           0EE2  1566 ;
                           0EE2  1567 ;       R3 -->  DISK [CLASS] DRIVER DDB
                           0EE2  1568 ;       R4 -->  DISK [CLASS] DRIVER DPT
                           0EE2  1569 ;       R5 -->  DISK [CLASS] DRIVER UCB
                           0EE2  1570 ;       R6 -->  RPB
```

```
                     0EE2  1571 ;       R7 -->  ADP FOR EITHER A REAL DISK OR A PORT
                     0EE2  1572 ;       R9 -->  PORT DRIVER DPT (IF PRESENT)
                     0EE2  1573 ;       R10-->  PORT DIRVER UCB (IF PRESENT)
                     0EE2  1574 ;
                     0EE2  1575 ; SPECIAL CODE FOR THE CONSOLE DEVICE
                     0EE2  1576 ;
 00000000'9F   16    0EE2  1577 70$:    JSB     @#INI$CONSOLE           ; Off to SYSLOA code for routine
                     0EE8  1578 ;
                     0EE8  1579 ; Check to see if we have been booted from a remote system (BOO$GQ_NODENAME
                     0EE8  1580 ; will be non-blank).  If so set up a system block for it, so that the
                     0EE8  1581 ; correct names can be generated.
                     0EE8  1582 ;
      01B0 8F   BB   0EE8  1583         PUSHR   #^M<R4,R5,R7,R8>        ; Save  registers
56 00000000'9F   DE   0EEC  1584         MOVAL   @#BOO$GB_NODENAME,R6    ; Address of remote node name
   66    08   20   3A  0EF3  1585         LOCC    #^A/ /,#8,(R6)          ; Find the end of the name
   58    08   50   C3  0EF7  1586         SUBL3   R0,#8,R8                ; Calculate name size
               55   13  0EFB  1587         BEQL    80$                     ; None, leave
   51    0060 8F   3C  0EFD  1588         MOVZWL  #SB$C_LENGTH,R1         ; Get size of system block
         0315      30  0F02  1589         BSBW    ALONONPAGED             ; Allocate it
         57   52   D0  0F05  1590         MOVL    R2,R7                   ; Copy to nonvolitle
   08 A7   51   B0  0F08  1591         MOVW    R1,SB$W_SIZE(R7)        ; Set in size field
      0760 8F   B0  0F0C  1592         MOVW    #DYN$C_SCS!<DYN$C_SCS_SB@8>,-
        0A A7       0F10  1593                  SB$B_TYPE(R7)           ; And type
        0C A7   9E  0F12  1594         MOVAB   SB$L_PBFL(R7),-         ; Init the path block listhead
        0C A7       0F15  1595                  SB$L_PBFL(R7)
        0C A7   9E  0F17  1596         MOVAB   SB$L_PBFL(R7),-
        10 A7       0F1A  1597                  SB$L_PBBL(R7)
      0000'CF   7D  0F1C  1598         MOVQ    W^BOO$GB_SYSTEMID,-
        18 A7       0F20  1599                  SB$B_SYSTEMID(R7)       ; Set in systemid
        1E A7   B4  0F22  1600         CLRW    SB$B_SYSTEMID+6(R7)     ; Just in case
53 00000000'9F   9E  0F25  1601         MOVAB   @#SCS$GQ_CONFIG,R3      ; Address of list head
      04 B3   67   0E  0F2C  1602         INSQUE  (R7),@4(R3)             ; Insert at tail
      44 A7   58   90  0F30  1603         MOVB    R8,SB$T_NODENAME(R7)    ; Insert size
0F    00   66   58   2C  0F34  1604         MOVC5   R8,(R6),#0,#15,-        ; Insert the name
            45 A7       0F39  1605                  SB$T_NODENAME+1(R7)
   00000054'9F   9E  0F3B  1606         MOVAB   @#SCS$GA_LOCALSB+ -
                53       0F41  1607                  SB$L_DDB-DDB$L_LINK,R3 ; Cover the DDB list head
         50   63   D0  0F42  1608         MOVL    DDB$L_LINK(R3),R0       ; R0 --> Boot DDB
      54 A7   50   D0  0F45  1609         MOVL    R0,SB$L_DDB(R7)         ; Insert it in the remote system block
               60   D0  0F49  1610         MOVL    DDB$L_LINK(R0),-
                63       0F4B  1611                  DDB$L_LINK(R3)          ; R1 --> Next DDB after boot DDB
               60   D4  0F4C  1612         CLRL    DDB$L_LINK(R0)          ; Show Boot DDB end of its list
      34 A0   57   D0  0F4E  1613         MOVL    R7,DDB$L_SB(R0)         ; Set back pointer to system block
                     0F52  1614 ;
                     0F52  1615 ; Now fill in correct equivalence string for SYS$DISK
                     0F52  1616 ;
56 00000000'FF   D0  0F52  1617 80$:    MOVL    @LNM$AL_HASHTBL,R6      ; Stuff address of data block
   51    1C   56   C1  0F59  1618         ADDL3   R6,#BDL_L_DISK_EQV-1,R1 ; Point to equivalence name (at "_")
   57    51   01   C1  0F5D  1619         ADDL3   #1,R1,R7                ; Copy at real name
         50   0F   D0  0F61  1620         MOVL    #15,R0                  ; Output length
      55   04 AE   D0  0F64  1621         MOVL    4(SP),R5                ; Recover R5
3C A5 00000200 8F   C8  0F68  1622         BISL    #DEV$M_NNM, -           ; Signal that system device uses the
                     0F70  1623                  UCB$L_DEVCHAR2(R5)      ; node$ddcn device name format.
         54   01   CE  0F70  1624         MNEGL   #1, R4                  ; Get NODE$ device name
   00000000'9F   16  0F73  1625         JSB     @#IOC$CVT_DEVNAM        ; Construct the name
 0C A6   51   01   A3  0F79  1626         SUBW3   #1,R1,BDL_W_DISK_EQ_SZ(R6) ; Update size
                     0F7E  1627 ;
```

```
                                0F7E  1628 ; Fill in correct equivalence string for SYS$SYSDEVICE
                                0F7E  1629 ;
     53    51 A6    9E          0F7E  1630         MOVAB   BDL_L_SYSD_EQV(R6),R3   ; Point to equiv name string (past '_')
           0C A6    B0          0F82  1631         MOVW    BDL_W_DISK_EQ_SZ(R6),-  ; Move size ( same as for SYS$DISK )
           40 A6                0F85  1632                 BDL_W_SYSD_EQ_SZ(R6)
              00'   E1          0F87  1633         BBC     S^#EXE$V_CONCEALED,-    ; Branch if not concealing dev name
08 00000000'GF                  0F89  1634                 G^EXE$GL_FLAGS,90$
   00000100 8F    C8          0F8F  1635         BISL2   #LNM$M_CONCEALED,-
           38 B6                0F95  1636                 @BDL_L_SYSD_AT_PTR(R6)  ; Mark concealed attribute
     63    67    51    28      0F97  1637 90$:    MOVC3   R1,(R7),(R3)            ; Copy device spec
           01B0 8F    BA        0F9B  1638         POPR    #^M<R4,R5,R7,R8>        ; Restore registers
                                0F9F  1639 ;
                                0F9F  1640 ; CONNECT UP CRB/IDB
                                0F9F  1641 ;
                                0F9F  1642         .ENABLE LSB
                 59    D5      0F9F  1643         TSTL    R9                      ; PORT DRIVER?
                 06    13      0FA1  1644         BEQL    10$                     ; NO
              7E 54    7D      0FA3  1645         MOVQ    R4,-(SP)                ; YES, SAVE DSK DPT/UCB ADDRESS
              54 59    7D      0FA6  1646         MOVQ    R9,R4                   ; TRANSFER PORT DPT/UCB ADDRESS
     24 A5    58    D0        0FA9  1647 10$:    MOVL    R8,UCB$L_CRB(R5)        ; CONNECT UCB TO CRB
           0C A8    B6        0FAD  1648         INCW    CRB$W_REFC(R8)          ; STEP COUNT OF CONNECTED UCB'S
     52    2C A8    D0        0FB0  1649         MOVL    CRB$L_INTD+VEC$L_IDB(R8),R2 ; GET ADDRESS OF IDB
                 12    13      0FB4  1650         BEQL    20$                     ; NONE
     0C A2       08    B0      0FB6  1651         MOVW    #8,IDB$W_UNITS(R2)      ; This IDB good for upto 8 units.
     50    54 A5    3C        0FBA  1652         MOVZWL  UCB$W_UNIT(R5),R0       ; Get devie unit number.
              50    08    D1  0FBE  1653         CMPL    #8,R0                   ; Will this unit fit in the IDB?
                    05    1B  0FC1  1654         BLEQU   20$                     ; Branch if unit will not fit.
                                0FC3  1655                                         ; Its a class driver disk and the
                                0FC3  1656                                         ; class driver doesn't use the IDB.
  18 A240    55    D0          0FC3  1657         MOVL    R5,IDB$L_UCBLST(R2)[R0] ; Else, plant UCB address in IDB.
                                0FC8  1658 ;
                                0FC8  1659 ; CONNECT DRIVER TO DEVICE DATA BASE AND PERFORM ANY NECESSARY RELOCATION.
                                0FC8  1660 ;
     00000000'GF    16          0FC8  1661 20$:    JSB     G^IOC$INITDRV           ; CONNECT DRIVER TO DATA BASE
        56    28 A5    D0      0FCE  1662         MOVL    UCB$L_DDB(R5),R6        ; GET DDB
0088 C5     0C A6    D0        0FD2  1663         MOVL    DDB$L_DDT(R6),UCB$L_DDT(R5) ; SET DDT IN UCB
64 A5     0800 8F    A8        0FD8  1664         BISW    #UCB$M_VALID,UCB$W_STS(R5) ; MUST FORCE ON SOFTWARE VOLUME VALID
                                0FDE  1665 ;
                                0FDE  1666 ; DITTO FOR CLASS DRIVER
                                0FDE  1667 ;
                 59    D5      0FDE  1668         TSTL    R9                      ; ANY CLASS DRIVER?
                 6E    13      0FE0  1669         BEQL    40$                     ; NO, LEAVE
              54 8E    7D      0FE2  1670         MOVQ    (SP)+,R4                ; YES, TRANSFER TO CORRECT REGISTER
                 002F  30      0FE5  1671         BSBW    ALLOC_CRB               ; GET A CRB
     24 A5       52    D0      0FE8  1672         MOVL    R2,UCB$L_CRB(R5)        ; CONNECT UCB TO CRB
           0C A2       B6      0FEC  1673         INCW    CRB$W_REFC(R2)          ; STEP COUNT OF CONNECTED UCB'S
     38 A2       57    D0      0FEF  1674         MOVL    R7,-                    ; FOR CLASS DRIVER, SAVE ADP POINTER IN
                                0FF3  1675                 CRB$L_INTD+VEC$L_ADP(R2); VEC SO THAT UNIBUS MAPPING ROUTINES
                                0FF3  1676                                         ; CAN WORK.
                 52    DD      0FF3  1677         PUSHL   R2                      ; Save CRB address.
              51 38    3C      0FF5  1678         MOVZWL  #IDB$K_LENGTH, R1       ; Set size to allocate.
                 021F  30      0FF8  1679         BSBW    ALONONPAGED             ; Allocate space for an IDB.
     08 A2       51    B0      0FFB  1680         MOVW    R1, IDB$W_SIZE(R2)      ; Set size of IDB.
     0A A2       09    90      0FFF  1681         MOVB    #DYN$C_IDB, -           ; Set type of IDB.
                       1003  1682                 IDB$B_TYPE(R2)
     0C A2       01    B0      1003  1683         MOVW    #1, IDB$W_UNITS(R2)     ; Set max units to one.
  62    0000'CF    9E          1007  1684         MOVAB   W^BOO$GB_SYSTEMID, -    ; Setup systemid address as CSR
```

```
                     100C  1685                       IDB$L_CSR(R2)           ; for the class driver.
          50 8ED0    100C  1686             POPL      R0                      ; Restore CRB address.
    2C A0    52  D0   100F  1687             MOVL      R2,-                    ; Store IDB address in CRB.
                     1013  1688                       CRB$L_INTD+VEC$L_IDB(R0)
          59  D4     1013  1689             CLRL      R9                      ; CLEAR A FLAG
          B1  11     1015  1690             BRB       20$                     ; CONNECT THIS DRIVER
                     1017  1691 ;
                     1017  1692 ; ALLOC_CRB - ALLOCATE AND PARTIALLY FILL A CRB BLOCK
                     1017  1693 ;
                     1017  1694 INI$ALLOC_CRB::
                     1017  1695 ALLOC_CRB:
  51   0048 8F  3C    1017  1696             MOVZWL    #CRB$C_LENGTH,R1        ; SET SIZE TO ALLOCATE
       01FB  30      101C  1697             BSBW      ALONONPAGED             ; ATTEMPT TO ALLOCATE IT
    08 A2  51  B0     101F  1698             MOVW      R1,CRB$W_SIZE(R2)       ; SET STRUCTURE SIZE
    0A A2  05  90     1023  1699             MOVB      #DYN$C_CRB,CRB$B_TYPE(R2); AND TYPE CODE
           62  DE     1027  1700             MOVAL     CRB$L_WQFL(R2),-
           62           1029  1701                       CRB$L_WQFL(R2)          ; BUILD WAIT QUEUE HEADER
           62  DE     102A  1702             MOVAL     CRB$L_WQBL(R2),-
       04 A2           102C  1703                       CRB$L_WQBL(R2)          ; FLINK AND BLINK
                05     102E  1704             RSB
                     102F  1705 ;
                     102F  1706 ; FIX_DRV_NAME - FIX UP THE DRIVER/DEVICE NAMES IN DDB/DDT
                     102F  1707 ;
                     102F  1708 FIX_DRV_NAME:
    50   20 A4  9A     102F  1709             MOVZBL    DPT$T_NAME(R4),R0       ; GET SIZE OF DRIVER NAME
           50  D6     1033  1710             INCL      R0                      ; BUMP TO INCLUDE COUNT
           3F  BB     1035  1711             PUSHR     #^M<R0,R1,R2,R3,R4,R5>  ; SAVE MOVC REGISTERS
    20 A4  50  28     1037  1712             MOVC3     R0,DPT$T_NAME(R4),-
       24 A3           103B  1713                        DDB$T_DRVNAME(R3)      ; SET DRIVER NAME IN DDB
           3F  BA     103D  1714             POPR      #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE MOVC REGISTERS
    52   14 A3  9E     103F  1715             MOVAB     DDB$T_NAME(R3),R2       ; GET ADDRESS OF DEVICE NAME
       82   03  90     1043  1716             MOVB      #3,(R2)+                ; SET COUNT FOR DDB NAME
    82   25 A3  B0     1046  1717             MOVW      DDB$T_DRVNAME+1(R3),(R2)+ ; ASSUME SAME AS START OF DRIVER
 62 55   41 8F  81     104A  1718             ADDB3     #^A/A7,R5,(R2)          ; SET CONTROLLER BASED ON ADAPTER COUNT
                05     104F  1719             RSB
                     1050  1720 ;
                     1050  1721 ; RESTORE PROPER IPL FOR POOL ALLOCATION
                     1050  1722 ;
00000000'GF  5B  D0    1050  1723 40$:        MOVL      R11,G^EXE$GL_NONPAGED   ; END OF INIT POOL ALLOCATION
                     1057  1724             .DISABLE LSB
                     1057  1725 ;
                     1057  1726 ; INIT DEVICE CONTROLLERS FOR DEVICES WITH RESIDENT DATABASES:
                     1057  1727 ;
                     1057  1728 INI_DEVICE:                                   ;
       5C  01  CE    1057  1729             MNEGL     #1,AP                   ; Set flag for INIT ALL
  00000000'GF  16    105A  1730             JSB       G^EXE$INIT_DEVICE
```

```
                                1060  1732          .SBTTL  MISCELLANEOUS CLEAN UP
                                1060  1733 ;
                                1060  1734 FINISH_UP:
          00000009'GF   16      1060  1735          JSB     G^INI$RDONLY            ;  AKE THE SYSTEM CODE READ ONLY
          00000000'GF   16      1066  1736          JSB     G^EXE$INIPROCREG        ; INIT PROCESSOR REGISTERS
          00000000'GF   16      106C  1737          JSB     G^MMG$ALLOCPFN          ; GET A MEMORY PAGE
    00000000'GF    50   D0      1072  1738          MOVL    R0,G^EXE$GL_BLAKHOLE    ; SAVE PFN - THIS IS THE RABBIT HOLE PAGE
          00000000'GF   D7      1079  1739          DECL    G^PFN$GL_PHYPGCNT       ; DECREASE NUMBER OF PAGES AVAILABLE
                                107F  1740
                                107F  1741 ; Allocate a PFN and a SPT entry for mount verification.  Build a PTE to map
                                107F  1742 ; the allocated PFN into system space.  Save the system virtual address of
                                107F  1743 ; the PTE in the global longword EXE$GL_SVAPTE.
                                107F  1744
          00000000'GF   16      107F  1745          JSB     G^MMG$ALLOCPFN          ; Get a memory page.
          00000000'GF   D7      1085  1746          DECL    G^PFN$GL_PHYPGCNT       ; Decrease number of pages available.
                  50    DD      108B  1747          PUSHL   R0                      ; Save allocated PFN value.
                01DA    30      108D  1748          BSBW    ALOSPT                  ; Get an SPTE.
50    00000000'FF40     DE      1090  1749          MOVAL   @MMG$GL_SPTBASE[R0], R0 ; Calculate the SVA of the SPTE.
  00000000'GF    50   D0      1098  1750          MOVL    R0, G^EXE$GL_SVAPTE     ; Save the SVA of the SPTE.
60    15    00   8E    F0      109F  1751          INSV    (SP)+, #0, -            ; Set PFN in allocated SPTE.
                                10A4  1752                  #PTE$S_PFN, (R0)
                                10A4  1753 ;
                                10A4  1754 ; ALLOCATE THE SYSTEM ERASE PATTERN BUFFER (EPB) AND THE PSUEDO
                                10A4  1755 ; PAGE TABLE (PPT) TO MAP IT.  THE EPB MUST BE PAGE ALIGNED.  BOTH
                                10A4  1756 ; PAGES ARE USED BY THE ERASE QIO FDT ROUTINE TO SPEED UP THE (FREQUENT)
                                10A4  1757 ; PROCESSING OF AN ERASE PATTERN OF 0 WRITTEN ON UP TO 127 BLOCKS.
                                10A4  1758 ;
                  22    10      10A4  1759          BSBB    1000$                   ; GET A PAGE OF MEMORY FOR SYSTEM EPB
    00000000'GF   50    D0      10A6  1760          MOVL    R0,G^EXE$GL_ERASEPB     ; SAVE ADDRESS OF SYSTEM EPB
                  51    DD      10AD  1761          PUSHL   R1                      ; SAVE SVAPTE
                  51    D4      10AF  1762          CLRL    R1                      ; SET PATTERN
                  3C    10      10B1  1763          BSBB    2000$                   ; PROPAGATE PATTERN TROUGH PAGE
                  13    10      10B3  1764          BSBB    1000$                   ; GET A PAGE OF MEMORY FOR SYSTEM PPT
    00000000'GF   50    D0      10B5  1765          MOVL    R0,G^EXE$GL_ERASEPPT    ; SAVE ADDRESS OF SYSTEM PPT
              51    9E    D0      10BC  1766          MOVL    @(SP)+,R1               ; GET PTE THAT MAPS SYSTEM EPB
51    04    1B    0F    F0      10BF  1767          INSV    #PRT$C_UR, #PTE$V_PROT, -
                                10C4  1768                  #PTE$S_PROT, R1         ; SET PPT PAGE PROTECTION TO UR
                  29    10      10C4  1769          BSBB    2000$                   ; PROPAGATE PTE THROUGH PPT
                  3A    11      10C6  1770          BRB     50$                     ; REJOIN NORMAL CODE PATH
                                10C8  1771
                                10C8  1772 ;
                                10C8  1773 ; LOCAL SUBROUTINE TO GET A PAGE (ALIGNED) OF MEMORY AND AN SPT
                                10C8  1774 ; AND TO MAP IT.  NO INPUT.  ON OUTPUT:
                                10C8  1775 ;        R0 = SYSTEM VIRTUAL ADDRESS (SVA) OF THE PAGE
                                10C8  1776 ;        R1 = SYSTEM VIRTUAL ADDRESS OF THE PTE (SVAPTE) THAT MAPS THE PAGE
                                10C8  1777 ;
          00000000'GF   16      10C8  1778 1000$:  JSB     G^MMG$ALLOCPFN          ; GET AND MAP A PAGE OF MEMORY
                  50    DD      10CE  1779          PUSHL   R0                      ; SAVE THE PFN
          00000000'GF   D7      10D0  1780          DECL    G^PFN$GL_PHYPGCNT       ; DECREMENT # OF FLUID PAGES
                0191    30      10D6  1781          BSBW    ALOSPT                  ; GET AN SPT TO MAP THE PFN TO A VA
51    00000000'FF40     DE      10D9  1782          MOVAL   @MMG$GL_SPTBASE[R0],R1  ; CALCULATE THE SVA OF THE SPTE
 61    15    00   8E    F0      10E1  1783          INSV    (SP)+,#0,#PTE$S_PFN,(R1); STORE THE PFN IN THE PTE
       50    50   09    78      10E6  1784          ASHL    #VA$V_VPN,R0,R0         ; CREATE THE SVA OF THE PAGE
         00  50   1F    E2      10EA  1785          BBSS    #VA$V_SYSTEM,R0,1010$   ; SET HIGH BIT TO INDICATE S0 ADDRESS
                      05    10EE  1786 1010$:  RSB
                                10EF  1787
                                10EF  1788 ;
```

```
                                10EF  1789 ; LOCAL SUBROUTINE TO PROPAGATE A LONGWORD THROUGH A PAGE.
                                10EF  1790 ;       R0 = ADDRESS OF PAGE
                                10EF  1791 ;       R1 = LONGWORD PATTERN
                                10EF  1792 ;
                          52    DD    10EF  1793 2000$:  PUSHL   R2                          ; SAVE R2
                00000080 8F     D0    10F1  1794         MOVL    #128,R2                     ; SET LOOP COUNTER
                      80  51    D0    10F8  1795 2010$:  MOVL    R1,(R0)+                    ; PUT PATTERN IN NEXT LONGWORD
                        FA 52   F5    10FB  1796         SOBGTR  R2,2010$                    ; BRANCH IF MORE
                      52    8E    D0    10FE  1797         MOVL    (SP)+,R2                    ; RESTORE R2
                                05    1101  1798         RSB
                                      1102  1799
          50    00000000'EF     3C    1102  1800 50$:    MOVZWL  SGN$GW_SYSDWSCT,R0          ; SYSTEM WORKING SET COUNT
     50   00000000'EF    50     C3    1109  1801         SUBL3   R0,SCH$GL_FREECNT,R0         ; SET AVAILABLE PAGE COUNT
00000000'EF 50    00000000'8F   C1    1111  1802         ADDL3   #<<MMG$A_SYS_END-MMG$AL_PGDCODEN>@-9>,R0,-
                                      111D  1803                 PFN$GL_PHYPGCNT              ; ADD INIT PAGES ALSO
                00000000'GF     D0    111D  1804         MOVL    G^SGN$GL_FREELIM,-
                00000000'GF           1123  1805                 G^SCH$GL_FREELIM             ; SET WORKING VALUE OF FREE LIMIT
                00000000'GF     C3    1128  1806         SUBL3   G^SCH$GL_FREELIM,-
          50    00000000'GF           112E  1807                 G^PFN$GL_PHYPGCNT,R0         ; COMPUTE MAX WS SIZE
          51    00000000'EF     3C    1134  1808         MOVZWL  MPW$GW_LOLIM,R1             ; SUBTRACT OUT FROM TOTAL
                      50  51    C2    113B  1809         SUBL    R1,R0
          50    00000000'EF     C2    113E  1810         SUBL    SCH$GL_FREELIM,R0           ; MINUS 2*FREELIM
          00000000'EF   50      D1    1145  1811         CMPL    R0,SGN$GL_MAXWSCNT          ; CHECK MAX WS COUNT FOR PHYS MEM
                        07      1E    114C  1812         BGEQU   60$                         ; BR IF OK
          00000000'EF   50      D0    114E  1813         MOVL    R0,SGN$GL_MAXWSCNT          ; OTHERWISE LIMIT WS SIZE TO AVAILABLE MEM
                                      1155  1814 60$:                                       ;
          7E    00000000'EF     9A    1155  1815         MOVZBL  SWP$GB_SHLP1PT,-(SP)        ; GET MANDATORY PAGE TABLES FOR SHELL
          51    00000000'EF     3C    115C  1816         MOVZWL  SGN$GW_MINWSCNT,R1          ; GET MINIMUM FLUID WORKING SET
          51    00000000'EF     C0    1163  1817         ADDL    SGN$GL_PHDPAGCT,R1          ; ADD FIXED PROCESS HEADER
                      51  8E    C0    116A  1818         ADDL    (SP)+,R1                    ; ADD NECESSARY PAGE TABLES
                      51  01'   C0    116D  1819         ADDL    S^#SWP$C_KSTACK+1,R1        ; ADD SPACE FOR KERNEL STACK AND POINTERS
                      51  50    D1    1170  1820         CMPL    R0,R1                       ; MIN WORKING SET MUST BE LESS THAN
                          0E    18    1173  1821         BGEQ    70$                         ; AVAILABLE PHYSICAL MEMORY
                51  EF93 CF    9E    1175  1822         MOVAB   W^NOPHYSMEM,R1              ; SET ADDRESS OF MESSAGE
                      5B    D4    117A  1823         CLRL    R11                         ; USE CONSOLE TERMINAL
                00000000'EF     16    117C  1824         JSB     EXE$OUTZSTRING              ; GIVE ERROR
                              00    1182  1825         HALT                                ; ***** FATAL ERROR *****
                                      1183  1826 70$:                                       ;
                      52    D4    1183  1827         CLRL    R2                          ; SIGNAL NO RETURN DATA EXPECTED
                    50  03    D0    1185  1828         MOVL    #CON$C_CLRWARM,R0           ; SIGNAL CLEAR WARMSTART INHIBIT FLAG
                00000000'GF     16    1188  1829         JSB     G^CON$SENDCONSCMD           ; CLEAR FLAG IN CONSOLE
                      52    D4    118E  1830         CLRL    R2                          ; SIGNAL NO RETURN DATA EXPECTED
                    50  04    D0    1190  1831         MOVL    #CON$C_CLRCOLD,R0           ; SIGNAL CLEAR COLD START INHIBIT FLAG
                00000000'GF     16    1193  1832         JSB     G^CON$SENDCONSCMD           ; CLEAR FLAG IN CONSOLE
                                      1199  1833 ;
                                      1199  1834 ;       MOVE A PIECE OF INIT INTO THE UNUSED POOL AND JUMP TO IT.
                                      1199  1835 ;       THIS SEGMENT OF CODE WILL RELEASE THE INIT PAGES TO THE FREELIST.
                                      1199  1836 ;
     56   00000004'EF     D0    1199  1837         MOVL    EXE$GL_NONPAGED+4,R6        ; GET ADDRESS OF FREE BLOCK
                56  08    C0    11A0  1838         ADDL    #8,R6                       ; SKIP HEADER
66  11B0'CF   0045'8F     28    11A3  1839         MOVC3   #INI_EXITSIZ,W^INI_EXITCODE,(R6)    ; COPY CODE TO POOL
                      7E    DC    11AB  1840         MOVPSL  -(SP)                       ; BUILD PC PCL PAIR
                      56    DD    11AD  1841         PUSHL   R6                          ; SET NEW PC
                            02    11AF  1842         REI                                 ; TRANSFER TO CODE IN POOL
                                      11B0  1843 ;
                                      11B0  1844 INI_EXITCODE:                            ; START OF EXIT CODE TO DESTROY INIT
     55   000011E9'EF     DE    11B0  1845         MOVAL   FREE,R5                     ; SET ADDRESS OF FREE MEMORY DESCRIPTOR
```

```
00000000'9F   9F   11B7  1846          PUSHAB  a#SCH$SCHED             ; SET EXIT TO SCHEDULER
                   11BD  1847                                           ; FALL THROUGH TO FILLSPT TO FREE
                   11BD  1848                                           ;  INIT PAGES;  FILLSPT RSB
                   11BD  1849                                           ;  EXITS TO SCH$SCHED
```

```
                                11BD  1851 ;
                                11BD  1852 ; FILL THE SPT AND RELEASE PREVIOUSLY MAPPED PAGES IF ANY
                                11BD  1853 ;
                                11BD  1854 ; INPUT:
                                11BD  1855 ;
                                11BD  1856 ;       R5 = DESCRIPTOR OF RANGE OF PAGES AND NEW PTE
                                11BD  1857 ;               0(R5) = 1ST VIRTUAL PAGE NUMBER
                                11BD  1858 ;               4(R5) = LAST + 1 VIRTUAL PAGE NUMBER
                                11BD  1859 ;               8(R5) = NEW PAGE TABLE ENTRY TO STORE
                                11BD  1860 ;
                                11BD  1861 ; OUTPUT:
                                11BD  1862 ;
                                11BD  1863 ;       R4 ALTERED
                                11BD  1864 ;
                                11BD  1865
                                11BD  1866 FILLSPT:
                          54    65    D0  11BD  1867          MOVL    (R5),R4
                00000000'9F     DD    DD  11C0  1868          PUSHL   @#MMG$GL_SPTBASE        ; GET BASE ADDRESS OF SPT
50  00 BE44     FFE00000 8F     CB    CB  11C6  1869 20$:     BICL3   #^C<PTE$M_PFN>,@(SP)[R4],R0 ; PFN FROM SPT ENTRY IF ANY
                      06    13    D0  11D0  1870          BEQL    40$                     ; BRANCH IF NONE THERE
                00000000'9F     16    D2  11D2  1871          JSB     @#MMG$DALLOCPFN         ; OTHERWISE DEALLOCATE IT
             00 BE44    08 A5    D0  11D8  1872 40$:     MOVL    8(R5),@(SP)[R4]         ; SET NEW SPT ENTRY
               E3 54    04 A5    F2  11DE  1873          AOBLSS  4(R5),R4,20$            ; REPEAT FOR EACH PAGE IN THE RANGE
                                11E3  1874          INVALID                         ; INVALIDATE THE TRANSLATION BUFFER
                         8E     D5  11E6  1875          TSTL    (SP)+                   ; CLEAN STACK
                                05  11E8  1876          RSB                             ; AND RETURN
                                11E9  1877
                    FFC00000'       11E9  1878 FREE:    .LONG   <INI_BASE-^X80000000>@-9        ; START OF INIT
                    FFC00000'       11ED  1879          .LONG   <MMG$A_SYS_END-^X80000000>@-9   ; END OF INIT
                    00000000        11F1  1880          .LONG   0                               ; NO ACCESS PTE
                    00000045        11F5  1881 INI_EXITSIZ=.-INI_EXITCODE               ; SIZE OF EXIT CODE
                                    11F5  1882 ;
                                    11F5  1883 ; SET THE PFN IN R0 RESIDENT
                                    11F5  1884 ;
                                    11F5  1885 ; INPUT:
                                    11F5  1886 ;
                                    11F5  1887 ;       R0 = PFN
                                    11F5  1888 ;
                                    11F5  1889 ; OUTPUT:
                                    11F5  1890 ;
                                    11F5  1891 ;       NONE
                                    11F5  1892 ;
                                    11F5  1893
                                    11F5  1894 SETRESIDENT:
                00000000'FF40   B6  11F5  1895          INCW    @PFN$AW_REFCNT[R0]      ; COUNT ONE REFERENCE
00000000'FF40   00000000'FF40   DE  11FC  1896          MOVAL   @MMG$GL_SPTBASE[R0],@PFN$AL_PTE[R0] ; BACK PTE POINTER
                00000000'FF40   07  90  1209  1897          MOVB    #PFN$C_ACTIVE,@PFN$AB_STATE[R0] ; PAGE IS ACTIVE
                00000000'FF40   01  90  1211  1898          MOVB    #PFN$C_SYSTEM,@PFN$AB_TYPE[R0] ; SYSTEM PAGE
                                05  1219  1899          RSB
```

```
                                121A  1901          .SBTTL  NONPAGED POOL ALLOCATION SUBROUTINES
                                121A  1902 ;+
                                121A  1903 ; ALONONPAGED CALLS EXE$ALONONPAGED TO ALLOCATE NON-PAGED POOL.  IF POOL
                                121A  1904 ; SPACE IS NOT SUFFICIENT TO CONTAIN THE REQUEST, A FATAL ERROR MESSAGE IS
                                121A  1905 ; GIVEN AND EXECUTION HALTS. THE PC ON THE TOP OF STACK WILL GIVE FURTHER
                                121A  1906 ; CONTEXT ABOUT THE ERROR.  THE ALLOCATED BLOCK WILL BE ZEROED.
                                121A  1907 ;
                                121A  1908 ; INPUT:
                                121A  1909 ;       R1 - SIZE OF REQUESTED BLOCK IN BYTES
                                121A  1910 ;
                                121A  1911 ; OUTPUTS:
                                121A  1912 ;       R0 - SUCCESS/FAILURE INDICATION
                                121A  1913 ;       R1 - ALLOCATED SIZE OF BLOCK
                                121A  1914 ;       R2 - ADDRESS OF BLOCK
                                121A  1915 ;-
                                121A  1916 INI$ALONONPAGED::                            ;
                                121A  1917 ALONONPAGED:                                 ;
                      51   DD   121A  1918          PUSHL   R1                        ; SAVE DESIRED ALLOCATION QUANTITY
           00000000'EF     16   121C  1919          JSB     EXE$ALONONPAGED           ; ATTEMPT TO ALLOCATE
                    2A 50  E8   1222  1920          BLBS    R0,10$                    ; CONTINUE IF ALLOCATED
                50 00000000'EF  7D   1225  1921          MOVQ    FIL$GQ_CACHE,R1           ; DEALLOCATE FIL$OPENFILE CACHE
                        1E  13   122C  1922          BEQL    5$                        ; BRANCH IF ALREADY GONE, FATAL
           00000000'EF     7C   122E  1923          CLRQ    FIL$GQ_CACHE              ; SAY IT IS GONE
                    50  52  D0   1234  1924          MOVL    R2,R0                     ; ADDRESS TO R0, SIZE IN R1
                        53  DD   1237  1925          PUSHL   R3                        ; SAVE THIS FROM DEALLOCATE
                53 00000004'EF  D0   1239  1926          MOVL    EXE$GL_NONPAGED+4,R3      ; ADDRESS OF FREE NON-PAGED POOL
           00000000'EF     16   1240  1927          JSB     EXE$DEALLOCATE            ; DEALLOCATE THE PIECE
                        08  BA   1246  1928          POPR    #^M<R3>                   ; RECOVER SAVED R3
                        02  BA   1248  1929          POPR    #^M<R1>                   ; GET ALLOCATION QUANTITY
                        CE  11   124A  1930          BRB     ALONONPAGED               ; AND TRY ALL OVER AGAIN
                        0F  10   124C  1931 5$:       BSBB    NOPOOLERR                 ; GIVE ERROR MESSAGE
                        00  124E  1932          HALT                              ; ***** FATAL ERROR *****
                    5E  04   C0   124F  1933 10$:      ADDL    #4,SP                     ; CLEAN OFF SAVED ALLOC SIZE
                        3F  BB   1252  1934          PUSHR   #^M<R0,R1,R2,R3,R4,R5>    ; SAVE MOVC REGISTERS
62  51  00  62  00   2C   1254  1935          MOVC5   #0,(R2),#0,R1,(R2)        ; ZERO FILL BLOCK
                        3F  BA   125A  1936          POPR    #^M<R0,R1,R2,R3,R4,R5>    ; RESTORE MOVC REGISTERS
                        05  125C  1937          RSB                               ; RETURN
                            125D  1938 ;
                            125D  1939 ; NOPOOLERR - SEND ERROR MESSAGE FOR INSUFFICIENT NON-PAGED POOL
                            125D  1940 ;
                            125D  1941 NOPOOLERR:                                   ;
                        5B  D4   125D  1942          CLRL    R11                       ; INDICATE CONSOLE TERMINAL
                51  EE7D CF   9E   125F  1943          MOVAB   NOSPACE,R1                ; SET ADDRESS OF ERROR MESSAGE
                 00000000'EF     17   1264  1944          JMP     EXE$OUTZSTRING            ; AND OUTPUT IT
```

```
                                126A  1946          .SBTTL  ALOSPT - ALLOCATE AND FILL SPT ENTRY FOR BUFFER WINDOW
                                126A  1947 ;+
                                126A  1948 ; ALOSPT ALLOCATES AN SPT SLOT TO BE USED FOR BUFFER OVERMAPPING WHEN
                                126A  1949 ; PERFORMING ECC CORRECTION OR OTHER SIMILAR EXCEPTIONAL I/O OPERATIONS.
                                126A  1950 ; THE SYSTEM PAGE TABLE ENTRY FOR THE ALLOCATED SYSTEM VIRTUAL PAGE NUMBER
                                126A  1951 ; WILL BE SET VALID, KERNEL WRITABLE BUT MAPPED TO A NON-EXISTENT PHYSICAL PAGE.
                                126A  1952 ; A FATAL ERROR MESSAGE WILL BE GIVEN IF UNABLE TO ALLOCATE SPT.
                                126A  1953 ;
                                126A  1954 ; INPUT:
                                126A  1955 ;       BOO$GL_SPTFREL - LOWEST FREE SYSTEM VPN
                                126A  1956 ;       BOO$GL_SPTFREH - HIGHEST FREE SYSTEM VPN
                                126A  1957 ;
                                126A  1958 ; OUTPUT:
                                126A  1959 ;       R0 - SYSTEM VPN ALLOCATED
                                126A  1960 ;       @MMG$GL_SPTBASE[R0] - PTE$M_VALID!PTE$M_PFN!PTE$C_KW
                                126A  1961 ;-
                                126A  1962 ALOSPT:                                              ;
50   00000000'EF    D0  126A  1963          MOVL    BOO$GL_SPTFREL,R0                       ; GET NEXT AVAILABLE SYSTEM VPN
00000000'EF    50   D1  1271  1964          CMPL    R0,BOO$GL_SPTFREH                       ; CHECK FOR REALLY AVAILABLE
               0E   15  1278  1965          BLEQ    10$                                     ; BRANCH IF YES
    51    EED3 CF   9E  127A  1966          MOVAB   NOSPT,R1                                ; SET ADDRESS OF ERROR MESSAGE
               5B   D4  127F  1967          CLRL    R11                                     ; INDICATE CONSOLE TERMINAL
     00000000'EF   16  1281  1968          JSB     EXE$OUTZSTRING                          ; OUTPUT ERROR MESSAGE
                   00  1287  1969          HALT                                            ; **** FATAL ERROR ****
     00000000'EF   D6  1288  1970 10$:     INCL    BOO$GL_SPTFREL                          ; MARK VPN ALLOCATED
      901FFFFF 8F  D0  128E  1971          MOVL    #<PTE$C_KW!PTE$M_VALID!PTE$M_PFN!PTE$C_KOWN>,-  ;
   00000000'FF40       1294  1972                  @MMG$GL_SPTBASE[R0]                     ; SET SPTE VALID, WRITABLE, NONEXISTENT PFN
                   05  129A  1973          RSB                                             ;
```

```
                     129B  1975          .SBTTL  RESIDENT PSECT CODE
                 00000000  1976          .PSECT  XDELTA,BYTE
                     0000  1977 ;
                     0000  1978 ;       INITIAL BREAKPOINT
                     0000  1979 ;
                     0000  1980 ; INPUT:
                     0000  1981 ;       NONE
                     0000  1982 ;
                     0000  1983 ; OUTPUT:
                     0000  1984 ;       CAUSES ENTRY TO DEBUGGER VIA BREAK POINT
                     0000  1985 ;
                     0000  1986 ; ***** WARNING DO NOT ALTER THIS ROUTINE, JUST A BPT AND RETURN.
                     0000  1987 ;
                     0000  1988
                     0000  1989 INI$BRK::
                  03 0000  1990          BPT                             ; STARTS AS BPT, CHANGED TO NOP
                  05 0001  1991          RSB                             ; RETURN
                     0002  1992
                     0002  1993 ;
                     0002  1994 ;       MASTER WAKE INTERRUPT, CAUSED BY SOFTWARE LEVEL 5 INTERRUPT:
                     0002  1995 ;
                 00000000  1996          .PSECT  $AEXENONPAGED,LONG
                     0000  1997
                     0000  1998          .ALIGN  LONG
                     0000  1999 INI$MASTERWAKE::                          ; SOFTWARE LEVEL 5 INTERRUPT
00000000'GF       16 0000  2000          JSB     G^INI$BRK               ; CALL BREAKPOINT SUBROUTINE
                  02 0006  2001          REI                             ; RETURN FROM INTERRUPT
```

```
                        0007  2003 ;
                        0007  2004 ; MAKE SYSTEM CODE READ ONLY, MAKE SYSTEM CODE READ/WRITE
                        0007  2005 ; USED BY XDELTA AND INIT.
                        0007  2006 ;
                        0007  2007 ; INPUTS:        NONE
                        0007  2008 ;
                        0007  2009 ; OUTPUTS:       ALL REGISTERS PRESERVED
                        0007  2010 ;
                 00000002  2011         .PSECT  XDELTA,BYTE
                        0002  2012
                        0002  2013         .ENABL  LSB
                        0002  2014 INI$WRITABLE::
                    0F  BB  0002  2015         PUSHR   #^M<R0,R1,R2,R3>        ; PRESERVE REGISTERS USED
                50  0E  9A  0004  2016         MOVZBL  #PRT$C_URKW,R0          ; PROTECTION TO USE
                    05  11  0007  2017         BRB     20$
                        0009  2018 INI$RDONLY::
                    0F  BB  0009  2019         PUSHR   #^M<R0,R1,R2,R3>        ; PRESERVE REGISTERS USED
                50  0F  9A  000B  2020         MOVZBL  #PRT$C_UR,R0            ; PROTECTION TO USE
                        000E  2021 20$:
             53 0036'CF  DE  000E  2022         MOVAL   W^INI_RDONLY_LIST,R3    ; GET START OF READ ONLY LIST
     18 00000000'EF  00' E0  0013  2023         BBS     S^#EXE$V_SYSWRTABL,EXE$GL_FLAGS,60$ ; BRANCH IF LEAVING SYSTEM WRITA
                51  83  7D  001B  2024         MOVQ    (R3)+,R1                ; GET A SET OF ADDRESS LIMITS
                    13  13  001E  2025         BEQL    60$                     ; DONE IF NULL
                        0020  2026 40$:
00000000'FF41  04  1B  50  F0  0020  2027         INSV    R0,#PTE$V_PROT,#PTE$S_PROT,@MMG$GL_SPTBASE[R1] ; SET PROTECTION
         FFF0 51  04  52  F1  002A  2028         ACBL    R2,#4,R1,40$            ; FOR EACH PAGE
                        0030  2029         INVALID                         ; INVALIDATE THE TRANSLATION BUFFER
                        0033  2030 60$:
                    0F  BA  0033  2031         POPR    #^M<R0,R1,R2,R3>        ; RESTORE SAVED REGISTERS
                    05  0035  2032         RSB                             ; AND RETURN TO THE CALLER
                        0036  2033         .DSABL  LSB                     ;
                        0036  2034 ;
                        0036  2035 ;        LIST OF READ ONLY SECTIONS IN THE RESIDENT EXEC
                        0036  2036 ;
                        0036  2037 INI_RDONLY_LIST:                         ;
                        0036  2038         .LIST   MEB
                        0036  2039         PURE    MMG$FRSTRONLY,<>        ; SECOND LONGWORD LOADED BY INIT
          FF000000'     0036               .LONG   <MMG$FRSTRONLY-<1@31>>@-7       ;
          00000000      003A               .LONG   0
                        003E  2040         PURE    MMG$AL_BEGDRIVE,MMG$AL_ENDDRIVE ;
          FF000000'     003E               .LONG   <MMG$AL_BEGDRIVE-<1@31>>@-7     ;
          00FFFFFC'     0042               .LONG   <MMG$AL_ENDDRIVE-^X80000200>@-7 ;
                        0046  2041 ;
                        0046  2042 ;        THE LIST MUST TERMINATE WITH THE FOLLOWING TWO DESCRIPTORS
                        0046  2043 ;
                        0046  2044 PGDCOD_LIM:                              ;
                        0046  2045         PURE    <>,MMG$AL_PGDCODEN      ; FIRST LONGWORD LOADED BY INIT
          00000000      0046               .LONG   0
          00FFFFFC'     004A               .LONG   <MMG$AL_PGDCODEN-^X80000200>@-7 ;
00000000  00000000      004E  2046         .LONG   0,0                     ; NULL DESCRIPTOR TO TERMINATE
                        0056  2047         .NLIST  MEB                     ;
                        0056  2048         .END    EXE$INIT
```

```
ADP$B_NUMBER                = 0000000B
ADP$L_LINK                  = 00000004
ADP$W_ADPTYPE               = 0000000E
ADP$W_TR                    = 0000000C
ALLOC_CRB                     00001017 R      04
ALONONPAGED                   0000121A R      04
ALOSPT                        0000126A R      04
ARC$M_CHAR_EMUL             = 00000010
ARC$M_CRC_EMUL              = 00000080
ARC$M_DCMC_EMUL             = 00000020
ARC$M_DFLT_EMUL             = 00000100
ARC$M_EDPC_EMUL             = 00000040
ARC$M_FFLT_EMUL             = 00000200
ARC$M_GFLT_EMUL             = 00000400
ARC$M_HFLT_EMUL             = 00000800
BADCONUCB                     0000017A R      04
BADDSKUCB                     000001A1 R      04
BADTTYDRV                     000001CA R      04
BAD_ADDRESS                   000001FF R      04
BAD_OPCODE                    00000248 R      04
BDL$GL_DISK_LOG               00000000 RG     04
BDL$L_SYSDLOG               = 00000034   G
BDL$S_CRELNM_ITMLST         = 00000074   G
BDL_L_DISK_AT_PTR           = 00000004
BDL_L_DISK_EQV              = 0000001D
BDL_L_DISK_EQ_PTR           = 00000010
BDL_L_SYSD_AT_PTR           = 00000038
BDL_L_SYSD_EQV              = 00000051
BDL_L_SYSD_EQ_PTR           = 00000044
BDL_W_DISK_EQ_SZ            = 0000000C
BDL_W_SYSD_EQ_SZ            = 00000040
BOO$GB_NODENAME               ********    X   04
BOO$GB_SYSTEMID               ********    X   04
BOO$GL_BOOTCB                 ********    X   04
BOO$GL_CLSLOA                 ********    X   04
BOO$GL_DEVNAME                ********    X   04
BOO$GL_DSKDRV                 ********    X   04
BOO$GL_ERAPATLOA              ********    X   04
BOO$GL_FPEMUL                 ********    X   04
BOO$GL_LRPMIN                 ********    X   04
BOO$GL_LRPSIZE                ********    X   04
BOO$GL_LRPSPLIT               ********    X   04
BOO$GL_MTACCESSLOA            ********    X   04
BOO$GL_NPAGEDYN               ********    X   04
BOO$GL_PRTDRV                 ********    X   04
BOO$GL_SCSLOA                 ********    X   04
BOO$GL_SPLITADR               ********    X   04
BOO$GL_SPTFREH                ********    X   04
BOO$GL_SPTFREL                ********    X   04
BOO$GL_SRPSPLIT               ********    X   04
BOO$GL_SYSLOA                 ********    X   04
BOO$GL_TRMDRV                 ********    X   04
BOO$GL_UCODE                  ********    X   04
BOO$GL_VAXEMUL                ********    X   04
BOO$GQ_FILCACHE               ********    X   04
BOO$GT_TOPSYS                 ********    X   04
BOO$L_CHECKSUM              = 00000000

BOO$L_SYS_MAP               = 00000014
BOO$L_SYS_VBN               = 0000000C
BTD$K_UDA                   = 00000011
CLASS_DDT                   = 00000010
CLASS_GETNXT                = 00000000
CLASS_PUTNXT                = 00000004
CLU$AL_LOAVEC                 ********    X   04
CLU$GL_ALLOCLS                ********    X   04
CLU_LOADCODE                  00000826 R      04
CON$C_CLRCOLD               = 00000004
CON$C_CLRWARM               = 00000003
CON$INIT_CTY                  ********    X   04
CON$SENDCONSCMD               ********    X   04
CR                          = 0000000D
CRB$B_TYPE                  = 0000000A
CRB$C_LENGTH                = 00000048
CRB$L_INTD                  = 00000024
CRB$L_WQBL                  = 00000004
CRB$L_WQFL                  = 00000000
CRB$W_REFC                  = 0000000C
CRB$W_SIZE                  = 00000008
DDB$B_TYPE                  = 0000000A
DDB$K_LENGTH                = 00000044
DDB$L_ALLOCLS               = 0000003C
DDB$L_DDT                   = 0000000C
DDB$L_LINK                  = 00000000
DDB$L_SB                    = 00000034
DDB$L_UCB                   = 00000004
DDB$T_DRVNAME               = 00000024
DDB$T_NAME                  = 00000014
DDB$W_SIZE                  = 00000008
DEV$M_NNM                   = 00000200
DPT$B_FLAGS                 = 0000000D
DPT$B_TYPE                  = 0000000A
DPT$L_FLINK                 = 00000000
DPT$T_NAME                  = 00000020
DPT$V_SVP                   = 00000001
DPT$W_UCBSIZE               = 0000000E
DPT$W_VECTOR                = 0000001E
DYN$C_CRB                   = 00000005
DYN$C_DDB                   = 00000006
DYN$C_DPT                   = 0000001E
DYN$C_HEADLEN               = 0000000C
DYN$C_IDB                   = 00000009
DYN$C_INIT                  = 00000063
DYN$C_LKID                  = 00000037
DYN$C_MPWMAP                = 00000004
DYN$C_ORB                   = 00000049
DYN$C_PCBVEC                = 00000001
DYN$C_PFL                   = 00000023
DYN$C_PHVEC                 = 00000002
DYN$C_PRCMAP                = 00000005
DYN$C_PTR                   = 00000025
DYN$C_RBM                   = 00000031
DYN$C_RSHT                  = 00000038
DYN$C_SCS                   = 00000060
DYN$C_SCS_SB                = 00000007
```

```
DYN$C_SWPMAP            =  00000003
DYN$C_UCB               =  00000010
DYN$C_WCB               =  00000012
END_INISPT                 000008D7 R      04
END_LOA                    00000852 R      04
ERAPAT_LOADCODE            000007ED R      04
EXE$ACVIOLAT               ********   X    04
EXE$ALONONPAGED            ********   X    04
EXE$AL_LOAVEC              ********   X    04
EXE$A_BOOPARAM             00000000 RG     03
EXE$BOOTCB_CHK             ********   X    04
EXE$BREAK                  ********   X    04
EXE$CMODEXECX              ********   X    04
EXE$CMODKRNLX              ********   X    04
EXE$DEALLOCATE             ********   X    04
EXE$ERAPAT_VEC             ********   X    04
EXE$GB_CPUDATA             ********   X    04
EXE$GB_CPUTYPE             ********   X    04
EXE$GL_ARCHFLAG            ********   X    04
EXE$GL_BLAKHOLE            ********   X    04
EXE$GL_BOOTCB              ********   X    04
EXE$GL_DEFFLAGS            ********   X    04
EXE$GL_ERASEPB             ********   X    04
EXE$GL_ERASEPPT            ********   X    04
EXE$GL_FLAGS               ********   X    04
EXE$GL_GPT                 ********   X    04
EXE$GL_INTSTK              ********   X    04
EXE$GL_INTSTKLM            ********   X    04
EXE$GL_NONPAGED            ********   X    04
EXE$GL_PAGED               ********   X    04
EXE$GL_RPB                 ********   X    04
EXE$GL_RTBITMAP            ********   X    04
EXE$GL_RTIMESPT            ********   X    04
EXE$GL_SCB                 ********   X    04
EXE$GL_SPLITADR            ********   X    04
EXE$GL_SVAPTE              ********   X    04
EXE$GL_SYSUCB              ********   X    04
EXE$GQ_BOOTCB_D            ********   X    04
EXE$GQ_TODCBASE            ********   X    04
EXE$INIPROCREG             ********   X    04
EXE$INIT                   000002A0 RG     04
EXE$INIT_DEVICE            ********   X    04
EXE$INI_TIMWAIT            ********   X    04
EXE$IPID_TO_EPID           ********   X    04
EXE$LINK_VEC               ********   X    04
EXE$MTACCESS_VEC           ********   X    04
EXE$OUTZSTRING             ********   X    04
EXE$RESTART                ********   X    04
EXE$ROPRAND                ********   X    04
EXE$TBIT                   ********   X    04
EXE$V_CONCEALED            ********   X    04
EXE$V_POOLPGING            ********   X    04
EXE$V_SSINHIBIT            ********   X    04
EXE$V_SYSPAGING            ********   X    04
EXE$V_SYSWRTABL            ********   X    05
FIL$GQ_CACHE               ********   X    04
FIL$GT_TOPSYS              ********   X    04

FILLSPT                    000011BD R      04
FINISH_UP                  00001060 R      04
FIX_DRV_NAME               0000102F R      04
FREE                       000011E9 R      04
HWTYPE                     000000C0 R      04
IDB$B_TYPE              =  0000000A
IDB$K_LENGTH            =  00000038
IDB$L_CSR               =  00000000
IDB$L_UCBLST            =  00000018
IDB$W_SIZE              =  00000008
IDB$W_UNITS             =  0000000C
INI$ALLOC_CRB              00001017 RG     04
INI$ALONONPAGED            0000121A RG     04
INI$BRK                    00000000 RG     05
INI$CONSOLE                ********   X    04
INI$MASTERWAKE             00000000 RG     06
INI$RDONLY                 00000009 RG     05
INI$WRITABLE               00000002 RG     05
INI_BASE                   00000000 R      04
INI_BOOTDEVIC              00000D48 R      04
INI_DEVICE                 00001057 R      04
INI_EXITCODE               000011B0 R      04
INI_EXITSIZ             =  00000045
INI_FREEMEM                0000051D R      04
INI_INTSTKLM               00000BD7 R      04
INI_IRP                    0000062E R      04
INI_LCKIDTBL               000008D7 R      04
INI_LOADCODE               000007E3 R      04
INI_LOG                    00000C8E R      04
INI_LRP                    00000672 R      04
INI_MISC                   00000B23 R      04
INI_MPW                    00000ADE R      04
INI_PAGDYN                 000005BC R      04
INI_PAGING                 000004CB R      04
INI_PFLVEC                 00000B8C R      04
INI_PHV                    00000A53 R      04
INI_PRCBITMAP              00000965 R      04
INI_PSTATE                 0000098C R      04
INI_RDONLY_LIST            00000036 R      05
INI_RESHTBL                00000928 R      04
INI_SPT                    00000852 R      04
INI_SRP                    000006D5 R      04
INI_SWAP                   00000AA4 R      04
INI_SYSPCB                 00000C17 R      04
INI_TTYDRV                 00000CBB R      04
IOC$CVT_DEVNAM             ********     X  04
IOC$GL_ADPLIST             ********     X  04
IOC$GL_DEVLIST             ********     X  04
IOC$GL_DPTLIST             ********     X  04
IOC$GL_IRPBL               ********     X  04
IOC$GL_IRPCNT              ********     X  04
IOC$GL_IRPMIN              ********     X  04
IOC$GL_IRPREM              ********     X  04
IOC$GL_LRPBL               ********     X  04
IOC$GL_LRPCNT              ********     X  04
IOC$GL_LRPMIN              ********     X  04
IOC$GL_LRPREM              ********     X  04
```

Symbol table

```
IOC$GL_LRPSIZE          ********    X   04
IOC$GL_LRPSPLIT         ********    X   04
IOC$GL_SRPBL            ********    X   04
IOC$GL_SRPCNT           ********    X   04
IOC$GL_SRPMIN           ********    X   04
IOC$GL_SRPREM           ********    X   04
IOC$GL_SRPSIZE          ********    X   04
IOC$GL_SRPSPLIT         ********    X   04
IOC$INITDRV             ********    X   04
IRP$C_LENGTH          = 000000C4
IRP$W_SIZE            = 00000008
LCK$GB_HTBLSHFT         ********    X   04
LCK$GB_MAXDEPTH         ********    X   04
LCK$GL_EXTRASTK         ********    X   04
LCK$GL_HASHTBL          ********    X   04
LCK$GL_HTBLCNT          ********    X   04
LCK$GL_HTBLSIZ          ********    X   04
LCK$GL_IDTBL            ********    X   04
LCK$GL_IDTBLSIZ         ********    X   04
LCK$GL_MAXID            ********    X   04
LCK$GL_NXTID            ********    X   04
LCK$GL_PRCMAP           ********    X   04
LF                    = 0000000A
LINK_INIT               00000838 R      04
LINK_INIT_RTN           00000841 R      04
LNM$AL_HASHTBL          ********    X   04
LNM$M_CONCEALED       = 00000100
LNM$M_TERMINAL        = 00000200
LNM$_ATTRIBUTES       = 00000003
LNM$_STRING           = 00000002
MB$DPT                  ********    X   04
MMG$ALLOCPFN            ********    X   04
MMG$AL_BEGDRIVE         ********    X   05
MMG$AL_ENDDRIVE         ********    X   04
MMG$AL_FIXUPTBL         ********    X   04
MMG$AL_PGDCODEN         ********    X   04
MMG$AL_SYSPCB           ********    X   04
MMG$A_SYS_END           ********    X   04
MMG$DALLOCPFN           ********    X   04
MMG$FRSTRONLY           ********    X   05
MMG$GL_GBLPAGFIL        ********    X   04
MMG$GL_GPTE             ********    X   04
MMG$GL_MAXPFN           ********    X   04
MMG$GL_MINPFN           ********    X   04
MMG$GL_NPAGEDYN         ********    X   04
MMG$GL_NULLPFL          ********    X   04
MMG$GL_PAGEDYN          ********    X   04
MMG$GL_PAGSWPVC         ********    X   04
MMG$GL_PGDCOD           ********    X   04
MMG$GL_SPTBASE          ********    X   04
MMG$GL_SYSLOA_BASE      ********    X   04
MMG$GL_SYSPHD           ********    X   04
MMG$GW_BIGPFN           ********    X   04
MMG$PAGEFAULT           ********    X   04
MPW$AL_PTE              ********    X   04
MPW$AW_PHVINDEX         ********    X   04
MPW$GW_HILIM            ********    X   04
MPW$GW_LOLIM            ********    X   04
MPW$GW_MPWPFC           ********    X   04
MSCP$K_EMD_OLD        = 00000000
MSCP$S_EU_CTYPE       = 00000004
MSCP$S_EU_DESIG       = 00000003
MSCP$S_EU_NO          = 00000008
MSCP$S_EU_SUBU        = 00000003
MSCP$V_EU_CTYPE       = 00000008
MSCP$V_EU_DESIG       = 0000000C
MSCP$V_EU_NO          = 00000000
MSCP$V_EU_SUBU        = 00000000
MTACCESS_LOADCODE       000007FD R      04
NL$DPT                  ********    X   04
NODEBUG                 0000043D R      04
NOPHYSMEM               0000010C R      04
NOPOCLERR               0000125D R      04
NOSPACE                 000000E0 R      04
NOSPT                   00000151 R      04
OP$DPT                  ********    X   04
OPA$GL_DDB              ********    X   04
OPA$UCBO                ********    X   04
ORB$B_FLAGS           = 0000000B
ORB$B_TYPE            = 0000000A
ORB$C_LENGTH          = 00000058
ORB$L_ACL_COUNT       = 00000028
ORB$L_ACL_DESC        = 0000002C
ORB$M_PROT_16         = 00000001
ORB$W_SIZE            = 00000008
OUTZ                    00000D06 R      04
PAGEDYN                 00000080 R      04
PCB$L_EPID            = 00000064
PCB$L_PHD             = 0000006C
PCB$L_PHYPCB          = 00000018
PCB$L_PID             = 00000060
PFL$L_VBN             = 00000010
PFL$L_WINDOW          = 0000000C
PFN$AB_STATE            ********    X   04
PFN$AB_TYPE             ********    X   04
PFN$AL_PTE              ********    X   04
PFN$AW_REFCNT           ********    X   04
PFN$C_ACTIVE          = 00000007
PFN$C_SYSTEM          = 00000001
PFN$GL_PHYPGCNT         ********    X   04
PGDCOD                  00000074 R      04
PGDCODBEG             = 00000074 R      04
PGDCODEND             = 00000078 R      04
PGDCOD_LIM              00000046 R      05
PHD$B_DFPFC           = 00000034
PHD$L_POBR            = 000000C8
PHD$L_POLRASTL        = 000000CC
PHD$L_PSTBASOFF       = 00000020
PHD$W_PSTLAST         = 0000002C
PHV$GL_PIXBAS           ********    X   04
PHV$GL_REFCBAS          ********    X   04
PR$_IPL               = 00000012
PR$_MAPEN             = 00000038
PR$_SBR               = 0000000C
```

Symbol table

PR$_SCBB	= 00000011	SB$W_SIZE	= 00000008				
PR$_SLR	= 0000000D	SCH$CHSE	********	X	04		
PR$_TBIA	= 00000039	SCH$GL_FREECNT	********	X	04		
PR$_TBIS	= 0000003A	SCH$GL_FREELIM	********	X	04		
PRT$C_UR	= 0000000F	SCH$GL_MAXPIX	********	X	04		
PRT$C_URKW	= 0000000E	SCH$GL_MFYLIM	********	X	04		
PTE$C_ERKW	= 30000000	SCH$GL_MFYLIMSV	********	X	04		
PTE$C_KOWN	= 00000000	SCH$GL_MFYLOLIM	********	X	04		
PTE$C_KW	= 10000000	SCH$GL_MFYLOSV	********	X	04		
PTE$C_UR	= 78000000	SCH$GL_NULLPCB	********	X	04		
PTE$M_PFN	= 001FFFFF	SCH$GL_PCBVEC	********	X	04		
PTE$M_TYP0	= 00400000	SCH$GL_PIXWIDTH	********	X	04		
PTE$M_TYP1	= 04000000	SCH$GL_SEQVEC	********	X	04		
PTE$M_VALID	= 80000000	SCH$GL_SWPPCB	********	X	04		
PTE$S_PFN	= 00000015	SCH$GW_PROCLIM	********	X	04		
PTE$S_PROT	= 00000004	SCH$SCHED	********	X	04		
PTE$V_PFN	= 00000000	SCS$AL_LOAVEC	********	X	04		
PTE$V_PROT	= 0000001B	SCS$GA_LOCALSB	********	X	04		
PTR$B_PTRTYPE	= 0000000B	SCS$GB_NODENAME	********	X	04		
PTR$B_TYPE	= 0000000A	SCS$GB_SYSTEMID	********	X	04		
PTR$K_LENGTH	= 00000010	SCS$GL_MCADR	********	X	04		
PTR$L_PTR0	= 00000010	SCS$GQ_CONFIG	********	X	04		
PTR$L_PTRCNT	= 0000000C	SCS_LOADCODE	0000080D	R	04		
PTR$W_SIZE	= 00000008	SEC$C_LENGTH	= 00000020				
RBM$B_TYPE	= 0000000A	SEC$L_REFCNT	= 00000018				
RBM$K_LENGTH	= 0000000C	SEC$L_VBN	= 00000010				
RBM$L_BITMAP	= 0000000C	SEC$L_VPXPFC	= 00000008				
RBM$L_FREECOUNT	= 00000004	SEC$L_WINDOW	= 0000000C				
RBM$L_STARTVPN	= 00000000	SETRESIDENT	000011F5	R	04		
RBM$W_SIZE	= 00000008	SGN$GB_SYSPFC	********	X	04		
RPB$B_CTRLLTR	= 00000108	SGN$GL_BALSETCT	********	X	04		
RPB$B_DEVTYP	= 00000066	SGN$GL_FREELIM	********	X	04		
RPB$L_BOOTR1	= 00000020	SGN$GL_GBLPAGFIL	********	X	04		
RPB$L_BOOTR5	= 00000030	SGN$GL_MAXWSCNT	********	X	04		
RPB$L_CHKSUM	= 00000008	SGN$GL_PAGEDYN	********	X	04		
RPB$L_ISP	= 000000A4	SGN$GL_PHDPAGCT	********	X	04		
RPB$L_RESTART	= 00000004	SGN$GL_SRPCNT	********	X	04		
RPB$L_RSTRTFLG	= 0000000C	SGN$GL_SRPMIN	********	X	04		
RPB$L_SBR	= 000000AC	SGN$GL_SRPSIZE	********	X	04		
RPB$L_SCBB	= 000000B0	SGN$GW_ISPPGCT	********	X	04		
RPB$L_SLR	= 000000B8	SGN$GW_MAXPRCCT	********	X	04		
RPB$Q_PFNMAP	= 00000044	SGN$GW_MINWSCNT	********	X	04		
RPB$V_DEBUG	= 00000001	SGN$GW_PAGFILCT	********	X	04		
RPB$V_INIBPT	= 00000002	SGN$GW_SWPFILCT	********	X	04		
RPB$W_UNIT	= 00000064	SGN$GW_SWPFILES	********	X	04		
SB$B_AVVERS	= 00000038	SGN$GW_SYSDWSCT	********	X	04		
SB$B_SYSTEMID	= 00000018	SWP$C_KSTACK	********	X	04		
SB$B_TYPE	= 0000000A	SWP$GB_SHLP1PT	********	X	04		
SB$C_LENGTH	= 00000060	SWP$GL_BALBASE	********	X	04		
SB$L_DDB	= 00000054	SWP$GL_BALSPT	********	X	04		
SB$L_PBBL	= 00000010	SWP$GL_MAP	********	X	04		
SB$L_PBFL	= 0000000C	SWP$GL_SHELLBAS	********	X	04		
SB$Q_SWINCARN	= 0000002C	SWP$GW_SWPINC	********	X	04		
SB$T_HWTYPE	= 00000034	SYS$GL_BOOTDDB	********	X	04		
SB$T_NODENAME	= 00000044	SYS$GL_BOOTUCB	********	X	04		
SB$T_SWTYPE	= 00000024	SYS$GT_ANNOUNCE	********	X	04		
SB$T_SWVERS	= 00000028	SYS$K_VERSION	********	X	04		

```
SYSWCB                          0000008C R      04
TT$V_SCOPE                    = 0000000C
TT2$M_AUTOBAUD                = 00000002
TTY$GL_DPT                      ********    X   04
UCB$L_ASTQFL                  = 0000000C
UCB$L_CRB                     = 00000024
UCB$L_DDB                     = 00000028
UCB$L_DDT                     = 00000088
UCB$L_DEVCHAR2                = 0000003C
UCB$L_IOQBL                   = 00000050
UCB$L_IOQFL                   = 0000004C
UCB$L_ORB                     = 0000001C
UCB$L_SVPN                    = 00000074
UCB$L_TT_CLASS                = 00000114
UCB$L_TT_DECHA1               = 000000C8
UCB$L_TT_DECHAR               = 000000C4
UCB$L_TT_GETNXT               = 0000010C
UCB$L_TT_PORT                 = 00000118
UCB$L_TT_PUTNXT               = 00000110
UCB$M_VACID                   = 00000800
UCB$W_MSCPUNIT                = 000000D4
UCB$W_SIZE                    = 00000008
UCB$W_STS                     = 00000064
UCB$W_UNIT                    = 00000054
VA$S_VPN                      = 00000015
VA$V_SYSTEM                   = 0000001F
VA$V_VPN                      = 00000009
VEC$L_ADP                     = 00000014
VEC$L_IDB                     = 00000008
WCB$B_ACCESS                  = 0000000B
WCB$B_TYPE                    = 0000000A
WCB$C_LENGTH                  = 00000030
WCB$C_MAP                     = 00000030
WCB$L_ORGUCB                  = 00000010
WCB$L_STVBN                   = 0000002C
WCB$M_CATHEDRAL               = 00000040
WCB$M_COMPLETE                = 00000020
WCB$M_NOTFCP                  = 00000004
WCB$M_READ                    = 00000001
WCB$W_NMAP                    = 00000016
WCB$W_SIZE                    = 00000008
XDELBPT                         ********    X   04
XDELTABASE                      00000000 R      02
XDELTBIT                        ********    X   04
XDEL_LOADBASE                   ********    X   04
XDS$GL_XESTRING                 ********    X   04
XDS$GL_XFSTRING                 ********    X   04
XDS$GT_LONG_PFN                 ********    X   04
XDS$GT_WORD_PFN                 ********    X   04
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
Z$DEBUGA	00000000 (0.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE
Z$INIT000	00000000 (0.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG
Z$INIT	0000129B (4763.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE
XDELTA	00000056 (86.)	05 (5.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
$AEXENONPAGED	00000007 (7.)	06 (6.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	29	00:00:00.07	00:00:00.73
Command processing	106	00:00:00.54	00:00:03.59
Pass 1	804	00:00:36.67	00:01:51.29
Symbol table sort	0	00:00:05.90	00:00:21.52
Pass 2	862	00:00:08.31	00:00:22.55
Symbol table output	23	00:00:00.43	00:00:00.97
Psect synopsis output	0	00:00:00.03	00:00:00.40
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1826	00:00:51.95	00:02:41.06

The working set limit was 2400 pages.
210057 bytes (411 pages) of virtual memory were used to buffer the intermediate code.
There were 200 pages of symbol table space allocated to hold 3704 non-local and 103 local symbols.
2048 source lines were read in Pass 1, producing 49 object records in Pass 2.
62 pages of virtual memory were used to define 61 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	44
_255DUA28:[SYSLIB]STARLET.MLB;2	12
TOTALS (all libraries)	56

3708 GETS were required to define 56 macros.

There were no errors, warnings or information messages.

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

FORKCNTRL
LIS

FILINIWCB
LIS

IMGDECODE
LIS

INIT
LIS

IOLOCK
LIS

IODEF
LIS

IOCIOPOST
LIS

GLOBALS
LIS

IMGMAPISD
LIS