```
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
```

FILEID**SHARE

```
  SSSSSSSS  HH      HH    AAAAAA    RRRRRRRR    EEEEEEEEEE
  SSSSSSSS  HH      HH    AAAAAA    RRRRRRRR    EEEEEEEEEE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
SS          HH      HH  AA      AA  RR      RR  EE
  SSSSSS    HHHHHHHHHH  AA      AA  RRRRRRRR    EEEEEEEE
  SSSSSS    HHHHHHHHHH  AA      AA  RRRRRRRR    EEEEEEEE
        SS  HH      HH  AAAAAAAAAA  RR  RR      EE
        SS  HH      HH  AAAAAAAAAA  RR  RR      EE
        SS  HH      HH  AA      AA  RR    RR    EE              ....
        SS  HH      HH  AA      AA  RR    RR    EE              ....
SSSSSSSS    HH      HH  AA      AA  RR      RR  EEEEEEEEEE      ....
SSSSSSSS    HH      HH  AA      AA  RR      RR  EEEEEEEEEE      ....

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  SHARE   SHARED MEMORY INITIALIZATION
0000     2          .IDENT  'V04-000'
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY:     SYSGEN
0000    31 ;
0000    32 ; ABSTRACT:     THIS MODULE INITIALIZES AND CONNECTS THE PROCESSOR TO
0000    33 ;               A PORT OF A MULTI-PORT (SHARED) MEMORY.
0000    34 ;
0000    35 ;
0000    36 ; ENVIRONMENT:  NATIVE/USER MODE, PRIVILEGED
0000    37 ;
0000    38 ; AUTHOR:       LEN KAWELL, CREATION DATE: 19-DEC-1978
0000    39 ;
0000    40 ; MODIFICATION HISTORY:
0000    41 ;
0000    42 ;       V03-007 KPL00100        Peter Lieberwirth       10-Feb-1984
0000    43 ;               Use longword format CONFREGL due to impending BI devices
0000    44 ;               having 16-bit device types.
0000    45 ;
0000    46 ;       V03-006 WHM00001        Bill Matthews           14-Dec-1983
0000    47 ;               Change references to ACF$B_CUNIT to ACF$W_CUNIT
0000    48 ;
0000    49 ;       V03-005 KDM42758        Kathleen D. Morse       04-Jan-1983
0000    50 ;               Minimize the shared memory structure quotas with their
0000    51 ;               maximums (e.g., SHD$W_MBXQUOTA with SHD$W_MBXMAX).
0000    52 ;               During a CONNECT, subtract one from the SHD$W_MBXQUOTA and
0000    53 ;               the SHD$W_CEFQUOTA counts for each structure owned by this port.
0000    54 ;
0000    55 ;
0000    56 ;--
```

```
                        58          .SBTTL  DECLARATIONS
                        59 ;
                        60 ; INCLUDE FILES:
                        61 ;
                        62
                        63 ;
                        64 ; MACROS:
                        65 ;
                        66
                        67 ;
                        68 ; PUT_OUTPUT - MACRO TO FORMAT AND PUT A MESSAGE TO SYS$OUTPUT
                        69 ;
                        70          .MACRO  PUT_OUTPUT MSG,ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
                        71          .SAVE   LSB
                        72          .PSECT  NONPAGED_DATA   RD,WRT,NOEXE,QUAD
                        73 $$DESC=.
                        74          .ASCID  \MSG\
                        75          .RESTORE
                        76
                        77          .IF NB ARG1                         ; IF FORMATTING NEEDED
                        78          MOVAB   -128(SP),SP                 ; ALLOCATE FORMAT BUFFER
                        79          PUSHL   SP                          ; CREATE BUFFER DESCRIPTOR
                        80          PUSHL   #128                        ;  ...
                        81          MOVL    SP,R0                       ; GET ADDR OF DESCRIPTOR
                        82          $FAO_S  $$DESC,(R0),(R0),-          ; FORMAT THE OUTPUT
                        83                  ARG1,ARG2,ARG3,ARG4,ARG5,ARG6,ARG7,ARG8
                        84          PUSHL   SP                          ; SET ADDR OF BUFFER DESC
                        85          CALLS   #1,G^LIB$PUT_OUTPUT         ; OUTPUT THE FORMATTED TEXT
                        86          MOVAB   128+8(SP),SP                ; DEALLOCATE BUFFER AND DESC
                        87
                        88          .IFF
                        89          PUSHAQ  $$DESC                      ; SET ADDR OF TEXT DESC
                        90          CALLS   #1,G^LIB$PUT_OUTPUT         ; OUTPUT THE TEXT
                        91          .ENDC
                        92
                        93          .ENDM   PUT_OUTPUT
                        94
                        95 ;
                        96 ; EQUATED SYMBOLS:
                        97 ;
                        98
08F0D180 00000000       99 INITLOCK_TIMOUT = 15*10*1000*1000            ; INITIALIZATION LOCK TIMEOUT TIME
02FAF080 00000000      100 INITPOLL_TIMOUT = 5*10*1000*1000             ; INITIALIZATION POLL TIMEOUT TIME
                       101
                       102 ;
                       103 ; SYSTEM DEFINITIONS
                       104 ;
                       105          $ACBDEF                            ; AST CONTROL BLOCKS
                       106          $ACFDEF                            ; CONFIGURATION CONTROL BLOCK
                       107          $ADPDEF                            ; NEXUS ADAPTER CONTROL BLOCKS
                       108          $CEBDEF                            ; COMMON EVENT FLAG BLOCKS
                       109          $DYNDEF                            ; DYNAMIC DATA STRUCTURE TYPE CODES
                       110          $GSDDEF                            ; GLOBAL SECTION DESCRIPTOR
                       111          $IPLDEF                            ; INTERRUPT PRIORITY LEVELS
                       112          $MBXDEF                            ; MAILBOX CONTROL BLOCK
                       113          $MPMDEF                            ; MULTIPORT MEMORY ADAPTER
                       114          $NDTDEF                            ; NEXUS DEVICE TYPES
```

```
                     0000   115          $PRDEF                          ; PROCESSOR REGISTERS
                     0000   116          $PRQDEF                         ; INTER-PROCESSOR REQUESTS
                     0000   117          $PTEDEF                         ; PAGE TABLE ENTRIES
                     0000   118          $RPBDEF                         ; RESTART PARAMETER BLOCK
                     0000   119          $RSNDEF                         ; RESOURCE NUMBERS
                     0000   120          $SHBDEF                         ; SHARED MEMORY CONTROL BLOCK
                     0000   121          $SHDDEF                         ; SHARED MEMORY DATAPAGE
                     0000   122          $SSDEF                          ; SYSTEM ERROR CODES
                     0000   123          $STSDEF                         ; STATUS CODES
                     0000   124          $SYSGMSGDEF                     ; SYSGEN MESSAGES
                     0000   125          $TPADEF                         ; TPARSE
                     0000   126          $VADEF                          ; VIRTUAL ADDRESSES
                     0000   127
                     0000   128 ;
                     0000   129 ; OWN STORAGE:
                     0000   130 ;
                  00000000   131          .PSECT  NONPAGED_DATA   RD,WRT,NOEXE,QUAD
                     0000   132
                     0000   133 SHR_VALUES:                              ; START OF QUALIFIER VALUES
                     0000   134 SHR_Q_MEMNAME:                          ; MEMORY NAME DESCRIPTOR
00000008             0000   135          .BLKL   2
                     0008   136 SHR_W_UNIT:                             ; MEMORY UNIT #
0000000A             0008   137          .BLKW   1
                     000A   138 SHR_W_GBLCNT:                          ; GLOBAL SECTION COUNT
0000000C             000A   139          .BLKW   1
                     000C   140 SHR_W_MBXCNT:                          ; MAILBOX COUNT
0000000E             000C   141          .BLKW   1
                     000E   142 SHR_W_CEFCNT:                          ; COMMON EVENT FLAG CLUSTER COUNT
00000010             000E   143          .BLKW   1
                     0010   144 SHR_W_GBLQUO:                          ; GLOBAL SECTION QUOTA FOR PORT
00000012             0010   145          .BLKW   1
                     0012   146 SHR_W_MBXQUO:                          ; MAILBOX QUOTA  FOR PORT
00000014             0012   147          .BLKW   1
                     0014   148 SHR_W_CEFQUO:                          ; COM EVT FLAG CLUSTER QUOTA FOR PORT
00000016             0014   149          .BLKW   1
                     0016   150 SHR_L_POOLBCNT:                        ; POOL BLOCK COUNT
0^00001A             0016   151          .BLKL   1
                     001A   152 SHR_L_POOLBSIZ:                        ; POOL BLOCK SIZE
0000001E             001A   153          .BLKL   1
                     001E   154 SHR_L_PRQCNT:                          ; INTER-PROCESSOR REQUEST COUNT
00000022             001E   155          .BLKL   1
                     0022   156 SHR_L_START:                            ; STARTING PFN
00000026             0022   157          .BLKL   1
                     0026   158 SHR_B_OPTIONS:                         ; COMMAND OPTIONS
00000027             0026   159          .BLKB   1
                     0027   160          _VIELD  SHR_OPT,0,<-            ;  OPTION DEFINITIONS
                     0027   161                  <INIT,,M>,-             ;   INITIALIZE MEMORY
                     0027   162                  >
                     0027   163
                     0027   164
                     0027   165 SHR_L_MEMSIZE:                          ; SIZE OF SHARED MEMORY (PAGES)
000000?B             0027   166          .BLKL   1
                     002B   167
                     002B   168 SHR_L_MEMPFN:                          ; STARTING PFN OF MEMORY
0000002F             002B   169          .BLKL   1
                     002F   170
                     002F   171 SHR_L_GSDSIZE:                         ; SIZE OF A GLOBAL SECTION DESC
```

```
                    00000033  002F   172          .BLKL   1
                              0033   173
                              0033   174 SHR_L_CEFSIZE:                      ; SIZE OF SHMCEB, MASTER COMM EVT BLOCK
                    00000037  0033   175          .BLKL   1
                              0037   176
                              0037   177 SHR_L_DATAPAGE:                     ; ADDRESS OF DATAPAGE
                    0000003B  0037   178          .BLKL   1
                              003B   179
                              003B   180 SHR_L_SHDPTE:                       ; ADDRESS OF DATAPAGE PTE
                    0000003F  003B   181          .BLKL   1
                              003F   182
                              003F   183 SHR_L_ADP:                          ; ADPATER CONTROL BLOCK ADDRESS
                    00000043  003F   184          .BLKL   1
                              0043   185
                              0043   186 SHR_T_MBDEVNAME:                    ; MAILBOX DEVICE NAME
                 42 42 4D 00' 0043   187          .ASCIC  /MBB/
                           03 0043
                              0047   188
                    00000000  0000   189          .PSECT  NONPAGED_CODE   RD,NOWRT,EXE,LONG ; PURE DATA SECTION
                              0000   190
                              0000   191 SHR_T_MBDRVNAME:                    ; MAILBOX DRIVER NAME
      52 45 56 49 52 44 58 42 4D 00' 0000   192          .ASCIC  /MBXDRIVER/
                           09 0000
                              000A   193
                              000A   194 ;
                              000A   195 ; AUTODIN-II POLYNOMIAL TABLE
                              000A   196 ;
                              000A   197
                              000A   198 AUTODIN:
26D930AC 3B6E20C8 1DB71064 00000000  000A   199          .LONG   ^000000000000,^003555610144,^007333420310,^004666230254
5005713C 4DB26158 6B6B51F4 76DC4190  001A   200          .LONG   ^016667040620,^015332650764,^011554460530,^012001270474
CB61938C D6D6A3E8 F00F9344 EDB88320  002A   201          .LONG   ^035556101440,^036003711504,^032665521750,^031330331614
BD..F21C A00AE278 86D3D2D4 9B64C2B0  003A   202          .LONG   ^023331141260,^020664751324,^024002561170,^027557371034
                              004A   203
```

```
                                    004A   205          .SBTTL  SHARE COMMAND QUALIFIER ACTION ROUTINES
                                    004A   206 ;++
                                    004A   207 ; FUNCTIONAL DESCRIPTION:
                                    004A   208 ;
                                    004A   209 ;       THESE TPARSE ACTION ROUTINES STORE THE SHARE COMMAND QUALIFIER
                                    004A   210 ;       VALUES AND CHECK THEIR VALIDITY.
                                    004A   211 ;
                                    004A   212 ;
                                    004A   213 ; CALLING SEQUENCE:
                                    004A   214 ;
                                    004A   215 ;       CALLED AS A TPARSE ACTION ROUTINE.
                                    004A   216 ;       (SEE THE RUN-TIME LIBRARY MANUAL FOR DETAILS)
                                    004A   217 ;
                                    004A   218 ; INPUTS:
                                    004A   219 ;
                                    004A   220 ;       STANDARD TPARSE PARAMETER BLOCK.
                                    004A   221 ;
                                    004A   222 ; OUTPUTS:
                                    004A   223 ;
                                    004A   224 ;       VALUES STORED FOR FUTURE USE BY SHARE COMMAND PROCESSING.
                                    004A   225 ;
                                    004A   226 ;       IF A VALUE IS INVALID, FAILURE IS RETURNED TO PRODUCE A SYNTAX
                                    004A   227 ;       ERROR AND STOP PROCESSING.
                                    004A   228 ;
                                    004A   229 ;
                                    004A   230 ;--
                                    004A   231          .PSECT  NONPAGED_CODE   RD,NOWRT,EXE,LONG
                                    004A   232          .DEFAULT DISPLACEMENT WORD      ; DEFAULT PC DISPLACEMENT
                                    004A   233
                                    004A   234
                                    004A   235 GEN$SHR_RESET::                          ; RESET QUALIFIER VALUES
                             0000   004A   236          .WORD   0                       ; ENTRY MASK
                 000A'CF    20   B0 004C   237          MOVW    #32,SHR_W_GBLCNT        ; SET DEFAULT GSD COUNT
                 000C'CF    20   B0 0051   238          MOVW    #32,SHR_W_MBXCNT        ; SET DEFAULT MAILBOX COUNT
                 000E'CF    20   B0 0056   239          MOVW    #32,SHR_W_CEFCNT        ; SET DEFAULT COM EV FLG CLUSTER COUNT
          0010'CF    7FFF 8F   B0 005B   240          MOVW    #32767,SHR_W_GBLQUO     ; SET DEFAULT GSD QUOTA FOR PORT
          0012'CF    7FFF 8F   B0 0062   241          MOVW    #32767,SHR_W_MBXQUO     ; SET DEFAULT MBX QUOTA FOR PORT
          0014'CF    7FFF 8F   B0 0069   242          MOVW    #32767,SHR_W_CEFQUO     ; SET DEFAULT CEF QUOTA FOR PORT
          001A'CF    0080 8F   3C 0070   243          MOVZWL  #128,SHR_L_POOLBSIZ     ; SET DEFAULT POOL BLOCK SIZE
          0016'CF    0080 8F   3C 0077   244          MOVZWL  #128,SHR_L_POOLBCNT     ; SET DEFAULT POOL BLOCK COUNT
          001E'CF    0040 8F   3C 007E   245          MOVZWL  #64,SHR_L_PRQCNT        ; SET DEFAULT PRQ COUNT
                     0022'CF   D4 0085   246          CLRL    SHR_L_START             ; SET DEFAULT STARTING PFN
                     0026'CF   94 0089   247          CLRB    SHR_B_OPTIONS           ; RESET ALL OPTIONS
                     50    01  D0 008D   248          MOVL    #1,R0                   ; SET SUCCESS
                               04 0090   249          RET                             ; RETURN
                                  0091   250
                                  0091   251 GEN$SHR_MEMNAME::                        ; SET THE MEMORY NAME
                             0000 0091   252          .WORD   0                       ; ENTRY MASK
                          0F   10 AC   B1 0093   253          CMPW    TPA$L_TOKENCNT(AP),#15  ; IS NAME TOO LONG?
                                  07   1E 0097   254          BGEQU   10$             ; BRANCH IF YES
                    0000'CF   10 AC   7D 0099   255          MOVQ    TPA$L_TOKENCNT(AP),SHR_Q_MEMNAME ; SET MEMORY NAME DESC
                               04   009F   256          RET                             ; EXIT
                                  00A0   257 10$:
                                  50   D4 00A0   258          CLRL    R0                      ; SET FAILURE
                               04   00A2   259          RET                             ; EXIT
                                  00A3   260
                                  00A3   261 GEN$SHR_UNIT::                          ; SET MEMORY UNIT #
```

```
                     0000  00A3   262          .WORD   0                               ; ENTRY MASK
0008'CF    1C AC      B0  00A5   263          MOVW    TPA$L_NUMBER(AP),SHR_W_UNIT ; SET MEMORY UNIT #
                      04  00AB   264          RET                                     ; EXIT
                          00AC   265
                          00AC   266 GEN$SHR_GBLCNT::                                ; SET GLOBAL SECTION COUNT
                     0000  00AC   267          .WORD   0                               ; ENTRY MASK
000A'CF    1C AC      B0  00AE   268          MOVW    TPA$L_NUMBER(AP),SHR_W_GBLCNT ; SET GLOBAL SECTION CNT
                      05  12  00B4   269          BNEQ    10$                             ; BRANCH IF AT LEAST 1
   000A'CF    01      B0  00B6   270          MOVW    #1,SHR_W_GBLCNT                 ; SET MINIMUM OF 1
                          00BB   271 10$:
                      04  00BB   272          RET                                     ; EXIT
                          00BC   273
                          00BC   274 GEN$SHR_MBXCNT::                                ; SET MAILBOX COUNT
                     0000  00BC   275          .WORD   0                               ; ENTRY MASK
000C'CF    1C AC      B0  00BE   276          MOVW    TPA$L_NUMBER(AP),SHR_W_MBXCNT ; SET MAILBOX COUNT
                      05  12  00C4   277          BNEQ    10$                             ; BRANCH IF AT LEAST 1
   000C'CF    01      B0  00C6   278          MOVW    #1,SHR_W_MBXCNT                 ; SET MINIMUM OF 1
                          00CB   279 10$:
                      04  00CB   280          RET                                     ; EXIT
                          00CC   281
                          00CC   282 GEN$SHR_CEFCNT::                                ; SET COMMON EVENT FLAG CLUSTER COUNT
                     0000  00CC   283          .WORD   0                               ; ENTRY MASK
000E'CF    1C AC      B0  00CE   284          MOVW    TPA$L_NUMBER(AP),SHR_W_CEFCNT ; SET COM EVT FLAG CLUSTER COUNT
                      05  12  00D4   285          BNEQ    10$                             ; BRANCH IF AT LEAST 1
   000E'CF    01      B0  00D6   286          MOVW    #1,SHR_W_CEFCNT                 ; SET MINIMUM OF 1
                          00DB   287 10$:
                      04  00DB   288          RET                                     ; EXIT
                          00DC   289
                          00DC   290 GEN$SHR_GBLMAX::                                ; SET PORT MAX GLOBAL SECTIONS
                     0000  00DC   291          .WORD   0                               ; ENTRY MASK
0010'CF    1C AC      B0  00DE   292          MOVW    TPA$L_NUMBER(AP),SHR_W_GBLQUO ; SET PORT MAX
                      04  00E4   293          RET                                     ; EXIT
                          00E5   294
                          00E5   295 GEN$SHR_MBXMAX::                                ; SET PORT MAX MAILBOXES
                     0000  00E5   296          .WORD   0                               ; ENTRY MASK
0012'CF    1C AC      B0  00E7   297          MOVW    TPA$L_NUMBER(AP),SHR_W_MBXQUO ; SET PORT MAX
                      04  00ED   298          RET                                     ; EXIT
                          00EE   299
                          00EE   300 GEN$SHR_CEFMAX::                                ; SET PORT MAX COM EVT FLG CLUSTERS
                     0000  00EE   301          .WORD   0                               ; ENTRY MASK
0014'CF    1C AC      B0  00F0   302          MOVW    TPA$L_NUMBER(AP),SHR_W_CEFQUO ; SET PORT MAX
                      04  00F6   303          RET                                     ; EXIT
                          00F7   304
                          00F7   305 GEN$SHR_POOLC::                                 ; SET POOL BLOCK COUNT
                     0000  00F7   306          .WORD   0                               ; ENTRY MASK
0016'CF    1C AC      D0  00F9   307          MOVL    TPA$L_NUMBER(AP),SHR_L_POOLBCNT ; SET POOL BLOCK COUNT
                      05  12  00FF   308          BNEQ    10$                             ; BRANCH IF NOT = 0
   0016'CF    01      D0  0101   309          MOVL    #1,SHR_L_POOLBCNT               ; SET MINIMUM OF 1
                          0106   310 10$:
                      04  0106   311          RET                                     ; EXIT
                          0107   312
                          0107   313 GEN$SHR_POOLS::                                 ; SET POOL BLOCK SIZE
                     0000  0107   314          .WORD   0                               ; ENTRY MASK
   50   001A'CF      DE  0109   315          MOVAL   SHR_L_POOLBSIZ,R0               ; GET ADDR OF SIZE BUFFER
     60    1C AC      D0  010E   316          MOVL    TPA$L_NUMBER(AP),(R0)           ; SET SPECIFIED POOL BLOCK SIZE
        51    1C      D0  0112   317          MOVL    #<ACB$L_KAST+4>,R1              ; GET MINIMUM SIZE (SIZE OF ACB)
        51    60      D1  0115   318          CMPL    (R0),R1                         ; IS SPECIFIED SIZE BIG ENOUGH?
```

```
                03   1E  0118   319          BGEQU   10$                     ; BRANCH IF YES
           50   51   D0  011A   320          MOVL    R1,R0                   ; ELSE, SET SIZE TO MINIMUM
                         011D   321 10$:
           60   07   C0  011D   322          ADDL    #^B111,(R0)             ; ROUND UP FOR QUADWORD ALIGNMENT
           60   07   CA  0120   323          BICL    #^B111,(R0)             ;  ...
           50   01   D0  0123   324          MOVL    #SS$_NORMAL,R0          ; RETURN SUCCESS
                     04  0126   325          RET                             ; EXIT
                         0127   326
                         0127   327 GEN$SHR_PRQCNT::                        ; SET INTER-PROCESSOR REQUEST BLOCK COUNT
                    0000 0127   328          .WORD   0                       ; ENTRY MASK
001E'CF    1C   AC   D0  0129   329          MOVL    TPA$L_NUMBER(AP),SHR_L_PRQCNT ; SET PRQ COUNT
                05   12  012F   330          BNEQ    10$                     ; BRANCH IF NOT = 0
   001E'CF      01   D0  0131   331          MOVL    #1,SHR_L_PRQCNT         ; SET MINIMUM OF 1
                         0136   332 10$:
                     04  0136   333          RET
                         0137   334
                         0137   335 GEN$SHR_START::                        ; SET START OF MEMORY
                    0000 0137   336          .WORD   0                       ; ENTRY MASK
0022'CF    1C   AC   D0  0139   337          MOVL    TPA$L_NUMBER(AP),SHR_L_START ; SET START OF MEMORY PFN
                     04  013F   338          RET                             ; EXIT
                         0140   339
                         0140   340 GEN$SHR_INIT::                         ; SET INIT OPTION
                    0000 0140   341          .WORD   0                       ; ENTRY MASK
   0026'CF      01   88  0142   342          BISB    #SHR_OPT_M_INIT,SHR_B_OPTIONS ; SET INIT OPTION
                     04  0147   343          RET                             ; EXIT
                         0148   344
```

```
                     0148   346          .SBTTL  SHARE COMMAND MAIN ACTION ROUTINE
                     0148   347 ;++
                     0148   348 ; FUNCTIONAL DESCRIPTION:
                     0148   349 ;
                     0148   350 ;       THIS IS THE MAIN SHARE COMMAND ACTION ROUTINE.  IT PERFORMS
                     0148   351 ;       ALL THE REAL WORK OF INITIALIZING AND/OR CONNECTING TO A SHARED
                     0148   352 ;       MEMORY.
                     0148   353 ;
                     0148   354 ;
                     0148   355 ; CALLING SEQUENCE:
                     0148   356 ;
                     0148   357 ;       CALLED AS A TPARSE ACTION ROUTINE.
                     0148   358 ;       (SEE THE RUN-TIME LIBRARY MANUAL FOR DETAILS)
                     0148   359 ;
                     0148   360 ; INPUTS:
                     0148   361 ;
                     0148   362 ;       STANDARD TPARSE PARAMETER BLOCK.
                     0148   363 ;
                     0148   364 ;       QUALIFIER VALUES ASSUMED TO BE STORED BY PREVIOUS ACTION
                     0148   365 ;       ROUTINES.
                     0148   366 ;
                     0148   367 ; OUTPUTS:
                     0148   368 ;
                     0148   369 ;       PROCESSOR CONNECTED TO THE SHARED MEMORY.  IF /INIT IS SPECIFIED,
                     0148   370 ;       THE MEMORY AND DATASTRUCTURES ARE ALSO INITIALIZED.
                     0148   371 ;
                     0148   372 ;--
                     0148   373
                     0148   374 GEN$SHARE::                                ; MAIN SHARE ACTION ROUTINE
                0000 0148   375          .WORD   0                          ; ENTRY MASK
                     014A   376
                     014A   377          $CMKRNL_S SHARE                    ; DO IT IN KERNEL MODE
           19 50  E9 0157   378          BLBC    R0,10$                     ; BRANCH IF FAILURE
                     015A   379          $CMEXEC_S SHOW_STRUCT              ; SHOW THE STRUCTURES
           09 50  E9 0167   380          BLBC    R0,10$                     ; BRANCH IF FAILURE
 083C'CF   00     FB 016A   381          CALLS   #0,LOADMBDRIVER            ; LOAD THE MAILBOX DRIVER
           01 50  E9 016F   382          BLBC    R0,10$                     ; BRANCH IF FAILURE
                  04 0172   383          RET                                ; EXIT
                     0173   384
                     0173   385 10$:
              02   F0 0173   386          INSV    #STS$K_ERROR,-             ; CONVERT STATUS TO ERROR
 50    03   00          0175   387                  #STS$V_SEVERITY,#STS$S_SEVERITY,R0
              50   DD 0178   388          PUSHL   R0                         ; SET ERROR
00000000'GF   01   FB 017A   389          CALLS   #1,G^LIB$SIGNAL            ; SIGNAL THE ERROR
           50 01   D0 0181   390          MOVL    #STS$K_SUCCESS,R0          ; SET SUCCESS FOR PARSER
                  04 0184   391          RET                                ; EXIT
                     0185   392
```

```
                                0185   394          .SBTTL  SHARE KERNEL ROUTINE
                                0185   395 ;++
                                0185   396 ;
                                0185   397 ; SHARE - KERNEL ROUTINE TO INIT AND CONNECT TO A SHARED MEMORY
                                0185   398 ;
                                0185   399 ; CALLING SEQUENCE:
                                0185   400 ;
                                0185   401 ;       $CMKRNL_S SHARE
                                0185   402 ;
                                0185   403 ; INPUTS:
                                0185   404 ;
                                0185   405 ;       SHARE COMMAND QUALIFIER VALUES STORED.
                                0185   406 ;
                                0185   407 ; OUPUTS:
                                0185   408 ;
                                0185   409 ;       R0 = SUCCESS OR FAILURE STATUS.
                                0185   410 ;
                                0185   411 ;       IF SUCCESS, MEMORY DATA STRUCTURES INITIALIZED (IF SO SPECIFIED)
                                0185   412 ;       AND MEMORY CONNECTED VIA THE SHARED MEMORY CONTROL BLOCK (SHB).
                                0185   413 ;
                                0185   414 ;--
                                0185   415 SHARE:                                       ; KERNEL ROUTINE
                           0FFC 0185   416          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; ENTRY MASK
                                0187   417 ;
                                0187   418 ; MAKE SURE THAT MA780 IS NOT BEING USED FOR MAIN MEMORY.
                                0187   419 ;
     50   007C810A 8F    D0   0187   420          MOVL    #SYSG$_SHMDBLUSE,R0     ; ASSUME ERROR
     54   00000000'GF    D0   018E   421          MOVL    G^EXE$GL_RPB,R4         ; GET ADDRESS OF RPB
 30  A4   00001800 8F    D3   0195   422          BITL    #<RPB$M_MPM ! RPB$M_USEMPM>,RPB$L_BOOTR5(R4) ; USED AS MAIN MEM?
                 6F    12   019D   423          BNEQ    ERR_EXIT                ; BRAND ON ERROR, IS USED AS MAIN MEM
                                019F   424 ;
                                019F   425 ; FIRST MAKE SURE THAT THE ADAPTER IS INITIALIZED AND CONNECTED
                                019F   426 ;
                      54    D4   019F   427          CLRL    R4                      ; INIT ADAPTER NUMBER
57   00000000'GF              D0   01A1   428          MOVL    G^EXE$GL_NUMNEXUS,R7    ; GET ADDRESS OF NUMBER OF NEXUSES
55   00000000'GF              D0   01A8   429          MOVL    G^EXE$GL_CONFREGL,R5    ; GET ADDRESS OF CONFREGL ARRAY
       52   6544   D0   01AF   430 10$:    MOVL    (R5)[R4],R2             ; GET ADAPTER TYPE CODE
                      25    13   01B3   431          BEQL    30$                     ; BRANCH IF NONE
      40 8F    52    91   01B5   432          CMPB    R2,#NDT$_MPM0           ; IS ADAPTER A MULTI-PORT MEMORY?
                      1F    1F   01B9   433          BLSSU   30$                     ; BRANCH IF NOT
      43 8F    52    91   01BB   434          CMPB    R2,#NDT$_MPM3           ; IS ADAPTER A MULTI-PORT MEMORY?
                      19    1A   01BF   435          BGTRU   30$                     ; BRANCH IF NOT
53   00000000'GF              D0   01C1   436          MOVL    G^IOC$GL_ADPLIST,R3     ; GET ADDRESS OF FIRST ADAPTER BLOCK
       54    0C A3    B1   01C8   437 20$:    CMPW    ADP$W_TR(R3),R4         ; IS THIS THE BLOCK FOR THE MEMORY?
                      0C    13   01CC   438          BEQL    30$                     ; BRANCH IF YES - NO NEED TO CREATE ONE
       53    04 A3    D0   01CE   439          MOVL    ADP$L_LINK(R3),R3       ; GET ADDRESS OF NEXT BLOCK
                      F4    12   01D2   440          BNEQ    20$                     ; BRANCH IF THERE IS ONE
      00000000'GF             16   01D4   441          JSB     G^INI$MPMADP            ; ELSE, CREATE AN ADAPTER CONTROL BLOCK
                              01DA   442 30$:
      D1 54    57    F2   01DA   443          AOBLSS  R7,R4,10$               ; INCREMENT ADAPTER NUMBER AND LOOP
                              01DE   444 ;
                              01DE   445 ; FIND THE SPECIFIED SHARED MEMORY UNIT AND GET ITS ADDRESS
                              01DE   446 ;
                              01DE   447 FIND_UNIT:
56   00000000'GF              D0   01DE   448          MOVL    G^IOC$GL_ADPLIST,R6     ; GET ADDR OF FIRST ADAPTER BLOCK
                      20    13   01E5   449          BEQL    30$                     ; BRANCH IF NONE
                              01E7   450 10$:                                        ; ADAPTER SEARCH LOOP
```

```
                OE A6   00'8F   91   01E7   451          CMPB    #AT$_MPM,ADP$W_ADPTYPE(R6) ; IS ADAPTER A MULTI-PORT?
                              13   12   01EC   452          BNEQ    20$                        ; BRANCH IF NOT
                          54  66   D0   01EE   453          MOVL    ADP$L_CSR(R6),R4           ; GET CSR OF ADAPTER
                       51  1C A4   D0   01F1   454          MOVL    MPM$L_MR(R4),R1            ; GET MAINTENENCE VALUE
                              0E   EF   01F5   455          EXTZV   #MPM$V_MR_UNIT,-           ; GET UNIT NUMBER
                    50  51    02             01F7   456                  #MPM$S_MR_UNIT,R1,R0       ;
                  0008'CF  50   B1   01FA   457          CMPW    R0,SHR_W_UNIT              ; IS IT DESIRED UNIT NUMBER?
                              16   13   01FF   458          BEQLU   INIT                       ; BRANCH IF YES
                                   0201   459 20$:
                     56   04 A6   D0   0201   460          MOVL    ADP$L_LINK(R6),R6          ; GET ADDR OF NEXT ADAPTER BLOCK
                              E0   12   0205   461          BNEQ    10$                        ; BRANCH IF ONE EXISTS
                                   0207   462 30$:
              50   007C8042 8F   D0   0207   463          MOVL    #SYSG$_NOSUCHMEM,R0        ; SET FAILURE
                                   020E   464 ERR_EXIT:
                              04   020E   465          RET                                ; EXIT
                                   020F   466 ;
                                   020F   467 ; SUBROUTINE USED BY INI$MPMADP TO ALLOCATE NON-PAGED POOL AND EXIT ROUTINE
                                   020F   468 ; CALL IF FAILURE
                                   020F   469 ;
                                   020F   470 INI$ALONONPAGED::                           ;
                        FDEE'  30   020F   471          BSBW    IOGEN$ALLOBLOCK            ; ALLOCATE A BLOCK
                       01 50   E9   0212   472          BLBC    R0,10$                     ; BRANCH IF FAILURE
                              05   0215   473          RSB                                ; ELSE, OK
                              04   0216   474 10$:     RET                                ; EXIT ROUTINE WITH STATUS
                                   0217   475 ;
                                   0217   476 ; INITIALIZE AND/OR CONNECT SHARED MEMORY
                                   0217   477 ;
                                   0217   478 INIT:
            003F'CF    56   D0   0217   479          MOVL    R6,SHR_L_ADP               ; SAVE ADP BLOCK ADDRESS
                    0062   30   021C   480          BSBW    CREATE_SHB                 ; CREATE SHARED MEM CONTROL BLCK
                  5B 50    E9   021F   481          BLBC    R0,EXIT                    ; BRANCH IF ERROR
                                   0222   482          SETIPL  #IPL$_HWCLK-1              ; SYNCHRONIZE LOCAL ACCESSORS
                    00A4   30   0225   483          BSBW    MAP_DATAPAGE               ; MAP THE DATAPAGE
                  52 50    E9   0228   484          BLBC    R0,EXIT                    ; BRANCH IF ERROR
                    012E   30   022B   485          BSBW    LOCK_DATAPAGE              ; LOCK THE DATAPAGE
                      00   E1   022E   486          BBC     #SHR_OPT_V_INIT,-          ; BRANCH IF /INIT NOT SPECIFIED
             1A 0026'CF         0230   487                  SHR_B_OPTIONS,CONNECT      ;
                    015E   30   0234   488          BSBW    CHECK_INIT                 ; CHECK IF OK TO INITIALIZE
                  14 50    E9   0237   489          BLBC    R0,CONNECT                 ; BRANCH IF NOT
                    01A9   30   023A   490          BSBW    INIT_DATAPAGE              ; INITIALIZE THE DATAPAGE
                  3A 50    E9   023D   491          BLBC    R0,UNLOCK_EXIT             ; BRANCH IF ERROR
                    031F   30   0240   492          BSBW    MAP_STRUCTURES             ; MAP THE OTHER DATA STRUCTURES
                  34 50    E9   0243   493          BLBC    R0,UNLOCK_EXIT             ; BRANCH IF ERROR
                    035E   30   0246   494          BSBW    INIT_STRUCTURES            ; INIT THE OTHER DATA STRUCTURES
                  2E 50    E9   0249   495          BLBC    R0,UNLOCK_EXIT             ; BRANCH IF ERROR
                      11   11   024C   496          BRB     CONNECTED                  ; INIT COMPLETED SUCCESSFULLY
                                   024E   497 ;
                                   024E   498 ; JUST CONNECT TO SHARED MEMORY
                                   024E   499 ;
                                   024E   500 CONNECT:                                ; CONNECT TO SHARED MEMORY
                      00   E0   024E   501          BBS     #SHB$V_CONNECT,-           ; BRANCH IF ALREADY CONNECTED
                 27 0B A5         0250   502                  SHB$B_FLAGS(R5),UNLOCK_EXIT
                    030C   30   0253   503          BSBW    MAP_STRUCTURES             ; MAP THE OTHER DATA STRUCTURES
                  21 50    E9   0256   504          BLBC    R0,UNLOCK_EXIT             ; BRANCH IF ERROR
                    0474   30   0259   505          BSBW    CONNECT_MEM                ; CONNECT TO DATA STRUCTURES
                  1B 50    E9   025C   506          BLBC    R0,UNLOCK_EXIT             ; BRANCH IF ERROR
                                   025F   507 CONNECTED:                              ; CONNECTED SUCCESSFULLY
```

```
                 0B A5    01    88  025F    508          BISB    #SHB$M_CONNECT,SHB$B_FLAGS(R5) ; SET MEMORY CONNECTED
                                   0263    509          DSBINT                                 ; LOCK OUT INTERRUPTS
                 54    1C A5    D0  0269    510          MOVL    SHB$L_ADP(R5),R4               ; GET ADDRESS OF ADP FOR THIS MA780
                 55    15 A5    9A  026D    511          MOVZBL  SHB$B_PORT(R5),R5              ; GET OWN PORT NUMBER
            00000000'GF    16    0271    512          JSB     G^MA$REQUEST                   ; FORCE INTERRUPT ON OWN PORT TO
                                   0277    513          ENBINT                                 ;  IMMEDIATELY PROCESS DANGLING PRQS
                                   027A    514 UNLOCK_EXIT:                                   ; UNLOCK DATAPAGE AND EXIT
                           0111    30  027A    515          BSBW    UNLOCK_DATAPAGE            ; UNLOCK DATAPAGE
                                   027D    516 EXIT:                                          ; EXIT KERNEL ROUTINE
                                   027D    517          SETIPL  #0                             ; RESTORE NORMAL IPL
                                   0280    518          RET                                    ; RETURN
                                   0281    519
```

```
                     0281   521          .SBTTL  CREATE SHARED MEMORY CONTROL BLOCK
                     0281   522 ;++
                     0281   523 ;
                     0281   524 ; CREATE_SHB - CREATE SHARED MEMORY CONTROL BLOCK
                     0281   525 ;
                     0281   526 ; THIS ROUTINE IS CALLED TO CREATE A SHARED MEMORY CONTROL BLOCK
                     0281   527 ; IN THE PROCESSOR'S LOCAL MEMORY POOL.
                     0281   528 ;
                     0281   529 ; INPUTS:
                     0281   530 ;
                     0281   531 ;       R4 = ADDRESS OF NEXUS CSR
                     0281   532 ;       R6 = ADAPTER CONTROL BLOCK ADDRESS
                     0281   533 ;
                     0281   534 ; OUTPUTS:
                     0281   535 ;
                     0281   536 ;       R0 = SUCCESS OR FAILURE STATUS
                     0281   537 ;       R5 = ADDRESS OF SHARED MEMORY CONTROL BLOCK
                     0281   538 ;
                     0281   539 ;       IF SHB FOR MEMORY DID NOT EXIST, IT IS CREATED AND LINKED
                     0281   540 ;       INTO SHB LIST (EXE$GL_SHBLIST).
                     0281   541 ;--
                     0281   542 CREATE_SHB:                              ; CREATE SHB
                     0281   543 ;
                     0281   544 ; CHECK IF SHARED MEMORY CONTROL BLOCK FOR THIS MEMORY ALREADY EXISTS.
                     0281   545 ;
                     0281   546          ASSUME  SHB$L_LINK EQ 0
          55 00000000'GF DE  0281   547          MOVAL   G^EXE$GL_SHBLIST,R5     ; GET ADDR OF SHB LIST
                     0288   548 10$:
               65    D5    0288   549          TSTL    SHB$L_LINK(R5)          ; IS THERE A NEXT SHB?
                  0C 13    028A   550          BEQL    20$                     ; BRANCH IF NOT
            55    65 D0    028C   551          MOVL    SHB$L_LINK(R5),R5       ; GET ADDR OF NEXT SHB
              14 A5 91     028F   552          CMPB    SHB$B_NEXUS(R5),-       ; IS THIS THE NEXUS?
                 0C A6             0292   553                  ADP$W_TR(R6)    ;
                  F2 12    0294   554          BNEQ    10$                     ; BRANCH IF NOT - TRY NEXT ONE
                  30 11    0296   555          BRB     30$                     ; ELSE - ALREADY EXISTS
                     0298   556 ;
                     0298   557 ; CREATE A SHARED MEMORY CONTROL BLOCK FOR THIS MEMORY PORT
                     0298   558 ;
                     0298   559 20$:                                         ;
            51    20 9A    0298   560          MOVZBL  #SHB$K_LENGTH,R1        ; SET SIZE OF SHB
                FD62' 30    029B   561          BSBW    IOGEN$ALLOBLOCK         ; ALLOCATE THE SHB
              2A 50 E9      029E   562          BLBC    R0,40$                  ; BRANCH IF FAILURE
            65    52 D0     02A1   563          MOVL    R2,SHB$L_LINK(R5)       ; SET FORWARD LINK TO SHB
            55    52 D0     02A4   564          MOVL    R2,R5                   ; SET ADDR OF SHB
                     02A7   565                                                 ; UNITIALIZED FIELDS ARE ZERO!
         08 A5    51 B0     02A7   566          MOVW    R1,SHB$W_SIZE(R5)       ; SET SIZE OF SHB IN SHB
         0A A5    2A 90     02AB   567          MOVB    #DYN$C_SHB,SHB$B_TYPE(R5) ; SET TYPE OF SHB IN SHB
         30 A6    55 D0     02AF   568          MOVL    R5,ADP$L_SHB(R6)        ; SET LINK TO SHB IN ADP
         1C A5    56 D0     02B3   569          MOVL    R6,SHB$L_ADP(R5)        ; SET LINK TO ADP IN SHB
      14 A5    0C A6 90     02B7   570          MOVB    ADP$W_TR(R6),SHB$B_NEXUS(R5) ; SET NEXUS NUMBER
            50    64 D0     02BC   571          MOVL    MPM$L_CSR(R4),R0        ; GET CSR
                  00 EF     02BF   572          EXTZV   #MPM$V_CSR_PORT,-       ; GET PORT NUMBER
       50    50    02              02C1   573                  #MPM$S_CSR_PORT,R0,R0
         15 A5    50 90     02C4   574          MOVB    R0,SHB$B_PORT(R5)       ; SET PORT NUMBER
                     02C8   575 30$:
            50    01 D0     02C8   576          MOVL    #1,R0                   ; SET SUCCESS
                     02CB   577 40$:
```

```
05  02CB   578          RSB                          ; RETURN
    02CC   579
```

```
                                02CC    581          .SBTTL  MAP THE DATAPAGE
                                02CC    582 ;++
                                02CC    583 ;
                                02CC    584 ; MAP_DATAPAGE - MAP SHARED MEMORY DATAPAGE (LAST PAGE IN MEMORY)
                                02CC    585 ;
                                02CC    586 ; THIS ROUTINE IS CALLED TO MAP THE SHARED MEMORY DATAPAGE INTO
                                02CC    587 ; THE SYSTEM VIRTUAL ADDRESS SPACE.
                                02CC    588 ;
                                02CC    589 ; INPUTS:
                                02CC    590 ;
                                02CC    591 ;       R4 = ADDR OF NEXUS CSR
                                02CC    592 ;       R5 = ADDR OF SHARED MEMORY CONTROL BLOCK (SHB)
                                02CC    593 ;
                                02CC    594 ;       IPL MUST BE IPL$_SYNCH.
                                02CC    595 ;
                                02CC    596 ; OUTPUTS:
                                02CC    597 ;
                                02CC    598 ;       R0 = SUCCESS OR FAILURE STATUS.
                                02CC    599 ;       R6 = ADDR OF DATAPAGE
                                02CC    600 ;
                                02CC    601 ;--
                                02CC    602 MAP_DATAPAGE:                               ; MAP THE DATAPAGE
      57    0C A4    D0        02CC    603          MOVL    MPM$L_INV(R4),R7           ; GET INVALIDATION REG VALUE
         0B    14    EF        02D0    604          EXTZV   #MPM$V_INV_STADR,#MPM$S_INV_STADR,- ;GET STARTING SBI
         50    57              02D3    605                  R7,R0                      ;LONGWORD ADDR<26:16> OF MEMORY
   50    50    09    78        02D5    606          ASHL    #16+2-VA$V_VPN,R0,R0       ; CONVERT TO A PFN
   0028'CF    50    D0        02D9    607          MOVL    R0,SHR_L_MEMPFN            ; SAVE MEMORY STARTING PFN
         03    10    EF        02DE    608          EXTZV   #MPM$V_INV_MEMSZ,#MPM$S_INV_MEMSZ,- ; GET MEMORY SIZE IN
         57    57              02E1    609                  R7,R7                      ; 256KB BOARD INCREMENTS
               57    D6        02E3    610          INCL    R7                         ; (0 = 1 BOARD)
57   00000200    8F    C4      02E5    611          MULL    #<256*1024>/512,R7         ; CONVERT TO PAGES
   0027'CF    57    D0        02EC    612          MOVL    R7,SHR_L_MEMSIZE           ; SAVE MEMORY SIZE
         57    50    C0        02F1    613          ADDL    R0,R7                      ; COMPUTE PFN OF LAST PAGE
               57    D7        02F4    614          DECL    R7                         ; ...
                               02F6    615
      56    04 A5    D0        02F6    616          MOVL    SHB$L_DATAPAGE(R5),R6      ; DATAPAGE ALREADY MAPPED?
               2C    12        02FA    617          BNEQ    20$                        ; BRANCH IF YES
                               02FC    618          ASSUME  SHD$K_LENGTH LE 512        ; ASSUME 1 PAGE
         51    01    D0        02FC    619          MOVL    #1,R1                      ; SET NUMBER PAGES
   00000000'GF    16        02FF    620          JSB     G^IOC$ALLOSPT              ; ALLOCATE A SPT ENTRY
            08 50    E8        0305    621          BLBS    R0,10$                     ; BRANCH IF SUCCESS
50   007C8022    8F    D0      0308    622          MOVL    #SYSG$_SPTFULL,R0          ; SET FAILURE STATUS
                     05        030F    623          RSB                                ; EXIT
                               0310    624 10$:
   56    52    09    78        0310    625          ASHL    #VA$V_VPN,R2,R6            ; CONVERT VPN TO VA
56   80000000    8F    C8      0314    626          BISL    #VA$M_SYSTEM,R6            ; ADD SYSTEM SPACE TO VA
      04 A5    56    D0        031B    627          MOVL    R6,SHB$L_DATAPAGE(R5)      ;  AND SAVE IN SHB
6342 57   B0000000    8F    C9 031F    628          BISL3   #<PTE$C_ERKW!PTE$M_VALID>,R7,- ; FILL-IN DATAPAGE SPT
                               0328    629                  (R3)[R2]                   ; ENTRY AND SET VALID
                               0328    630 20$:
50   00000000'GF    D0        0328    631          MOVL    G^EXE$GL_RPB,R0            ; GET ADDRESS OF RPB
50   000000BC    8F    C0      032F    632          ADDL    #RPB$L_MEMDSC,R0           ; POINT TO FIRST MEMORY DESCRIPTOR
                               0336    633 30$:
   14 A5    03 A0    91        0336    634          CMPB    3(R0),SHB$B_NEXUS(R5)      ; DOES MA780 TR NUMBER MATCH THIS DSC?
               16    13        033B    635          BEQL    40$                        ; BR IF FOUND THE MEMORY DESCRIPTOR
         50    08    C0        033D    636          ADDL    #8,R0                      ; POINT TO NEXT DESCRIPTOR
               60    D5        0340    637          TSTL    (R0)                       ; IS THERE ANOTHER MEMORY TO CHECK?
```

```
                     F2   12  0342    638          BNEQ    30$                     ; BR IF THERE IS ANOTHER VALID DSC
                             0344    639 ;
                             0344    640 ; NO MEMORY DESCRIPTOR WAS FOUND FOR THIS MA780.  THEREFORE, IT WAS
                             0344    641 ; PROBABLY POWERED UP AFTER THE SYSTEM WAS BOOTED.  A MEMORY DESCRIPTOR
                             0344    642 ; IN THE RPB MUST BE CREATED SO THAT THE MA780 PAGES WILL GET WRITTEN
                             0344    643 ; TO THE DUMP FILE DURING A BUGCHECK.
                             0344    644 ;
      80   0027'CF    D0     0344    645          MOVL    SHR_L_MEMSIZE,(R0)+     ; SET # OF PAGES OF MEMORY
    FF A0    14 A5    90     0349    646          MOVB    SHB$B_NEXUS(R5),-1(R0)  ; SET TR # OF MEMORY
      60   002B'CF    D0     034E    647          MOVL    SHR_L_MEMPFN,(R0)       ; SET STARTING PHYS ADR OF MEMORY
                             0353    648 40$:
           50    01    D0     0353    649          MOVL    #1,R0                   ; SET SUCCESS
     0037'CF    56    D0     0356    650          MOVL    R6,SHR_L_DATAPAGE       ; SAVE ADDRESS OF DATAPAGE
                       05    035B    651          RSB                             ; RETURN
                             035C    652
```

```
                     035C   654          .SBTTL  LOCK/UNLOCK THE DATAPAGE
                     035C   655 ;++
                     035C   656 ;
                     035C   657 ; LOCK_DATAPAGE - LOCK THE DATAPAGE FOR INITIALIZATION/CONNECTION
                     035C   658 ; UNLOCK_DATAPAGE - UNLOCK THE DATAPAGE
                     035C   659 ;
                     035C   660 ; THE INIT FLAG IS CLEAR WHEN IT IS LOCKED FOR INITIALIZATION.  THIS IS
                     035C   661 ; BECAUSE THE MEMORY IS INITIALIZED TO ALL 1'S WHEN IT IS POWERED ON AND
                     035C   662 ; THE COMPLETE TIMEOUT WOULD HAVE TO ELAPSE EVERYTIME A NEWLY POWERED-ON
                     035C   663 ; MEMORY WAS INITIALIZED.  TO AVOID THIS, THE SENSE OF THE LOCK IS
                     035C   664 ; REVERSED.
                     035C   665 ;
                     035C   666 ; INPUTS:
                     035C   667 ;
                     035C   668 ;       R5 = ADDRESS OF SHARED MEMORY CONTROL BLOCK (SHB)
                     035C   669 ;       R6 = ADDRESS OF DATAPAGE
                     035C   670 ;
                     035C   671 ;       IPL LESS THAN IPL$_HWCLK SO TIME CAN BE UPDATED.
                     035C   672 ;
                     035C   673 ; OUTPUTS:
                     035C   674 ;       THE INIT FLAG IS CLEARED OR SET, INDICATING A PORT IS CURRENTLY
                     035C   675 ;       INITIALIZING/CONNECTING OR DONE INITIALIZING CONNECTING, RESPECTIVELY.
                     035C   676 ;
                     035C   677 LOCK_DATAPAGE:                            ; LOCK DATAPAGE INIT LOCK
50   00000000'GF  7D  035C   678          MOVQ    G^EXE$GQ_SYSTIME,R0     ; GET CURRENT SYSTEM TIME
50   08F0D180 8F  C0  0363   679          ADDL    #INITLOCK_TIMOUT,R0     ; COMPUTE TIMEOUT TIME
          51      00  D8  036A   680          ADWC    #0,R1                   ;
 02 009F C6      00  E7  036D   681 10$:     BBCCI   #SHD$V_INITLCK,SHD$B_FLAGS(R6),20$
                 12  11  0373   682          BRB     30$
00000004'GF      51  D1  0375   683 20$:     CMPL    R1,G^EXE$GQ_SYSTIME+4   ; TIMEOUT?
                 EF  1A  037C   684          BGTRU   10$                     ; IF GTRU, NO
00000000'GF      50  D1  037E   685          CMPL    R0,G^EXE$GQ_SYSTIME     ; TIMEOUT?
                 E6  1A  0385   686          BGTRU   10$                     ; IF GTRU, NO
                     0387   687 30$:
 009D C6   15    A5  90  0387   688          MOVB    SHB$B_PORT(R5),SHD$B_INITLCK(R6) ; SET LOCKING PORT NUMBER
                     05  038D   689          RSB                             ;
                     038E   690
                     038E   691 UNLOCK_DATAPAGE:                          ; UNLOCK THE DATAPAGE INIT LOCK
 00 009F C6      00  E6  038E   692          BBSSI   #SHD$V_INITLCK,SHD$B_FLAGS(R6),10$
                     0394   693 10$:
                     05  0394   694          RSB
                     0395   695
```

```
                     0395   697          .SBTTL  CHECK IF MEMORY CAN BE INITIALIZED
                     0395   698 ;++
                     0395   699 ;
                     0395   700 ; CHECK_INIT - CHECK IF MEMORY CAN BE INITIALIZED
                     0395   701 ;
                     0395   702 ; THIS ROUTINE IS CALLED TO CHECK THAT NO OTHER PORTS ARE USING THE
                     0395   703 ; THE MEMORY AND IT IS ALRIGHT TO INTIALIZE IT.
                     0395   704 ;
                     0395   705 ; INPUTS:
                     0395   706 ;
                     0395   707 ;       R4 = ADDR OF NEXUS CSR
                     0395   708 ;       R5 = ADDR OF SHB
                     0395   709 ;       R6 = ADDR OF DATAPAGE (SHD)
                     0395   710 ;
                     0395   711 ; OUTPUTS:
                     0395   712 ;
                     0395   713 ;       R0 = SUCCESS IF MEMORY CAN BE INITIALIZED.
                     0395   714 ;
                     0395   715 ;       THIS PORT'S REFERENCE COUNT TO THE MEMORY IS CHECKED, IF IT IS NON-ZERO,
                     0395   716 ;       THE MEMORY CAN'T BE INITIALIZED.
                     0395   717 ;
                     0395   718 ;       THE OTHER PORTS ARE POLLED TO SEE IF THEY ARE CONNECTED TO THE MEMORY
                     0395   719 ;       BY CLEARING A POLLING MASK AND INTERRUPTING ALL THE PORTS.  IF A PORT
                     0395   720 ;       IS CONNECTED, IT WILL SET ITS POLLING FLAG, INDICATING THE MEMORY
                     0395   721 ;       SHOULD NOT BE INITIALIZED.  IF THE TIMEOUT EXPIRES AND NO PORT HAS
                     0395   722 ;       SET A POLLING FLAG, IT IS OK TO INITIALIZE.
                     0395   723 ;
                     0395   724 ;--
                     0395   725 CHECK_INIT:                                ; CHECK IF MEMORY CAN BE INITED
 OC A5   D5          0395   726          TSTL    SHB$L_REFCNT(R5)         ; ANY REFERENCES TO MEMORY?
    49   12          0398   727          BNEQ    NO_INIT                  ; BRANCH IF YES
                     039A   728 ;
                     039A   729 ; POLL OTHER PORTS TO SEE IF THEY ARE CONNECTED TO THE MEMORY
                     039A   730 ;
                     039A   731 POLL:
        0070 8F      BB     039A   732          PUSHR   #^M<R4,R5,R6>            ; SAVE REGISTERS
        00A6 C6      B4     039E   733          CLRW    SHD$W_POLL(R6)           ; CLEAR POLLING FLAGS
  54     1C A5      D0     03A2   734          MOVL    SHB$L_ADP(R5),R4         ; SET ADDRESS OF ADAPTER CONTROL BLOCK
             55      D4     03A6   735          CLRL    R5                       ; INIT PORT NUMBER
 00000000'GF          16     03A8   736 5$:      JSB     G^MA$REQUEST             ; WAKEUP THE PROCESSOR AT THE PORT
  F6 55              04     F2     03AE   737          AOBLSS  #MPM$C_PORTS,R5,5$       ; INCREMENT PORT NUMBER AND LOOP
        0070 8F      BA     03B2   738          POPR    #^M<R4,R5,R6>            ; RESTORE REGISTERS
                     03B6   739
50   00000000'GF     7D     03B6   740          MOVQ    G^EXE$GQ_SYSTIME,R0      ; GET CURRENT SYSTEM TIME
50   02FAF080 8F     C0     03BD   741          ADDL    #INITPOLL_TIMOUT,R0      ; COMPUTE TIMEOUT TIME
        51   00      D8     03C4   742          ADWC    #0,R1                    ; ...
        00A6 C6      B5     03C7   743 10$:     TSTW    SHD$W_POLL(R6)           ; ANY PORT ACTIVE?
             16      12     03CB   744          BNEQ    NO_INIT                  ; IF NEQ, YES - CAN'T INITIALIZE
00000004'GF  51      D1     03CD   745 20$:     CMPL    R1,G^EXE$GQ_SYSTIME+4    ; TIMEOUT?
             F1      1A     03D4   746          BGTRU   10$                      ; IF GTRU, NO
00000000'GF  50      D1     03D6   747          CMPL    R0,G^EXE$GQ_SYSTIME      ; TIMEOUT?
             E8      1A     03DD   748          BGTRU   10$                      ; IF GTRU, NO
                     03DF   749 30$:
        50   01      D0     03DF   750          MOVL    #1,R0                    ; OK TO INITIALIZE
                     05     03E2   751          RSB                              ; RETURN
                     03E3   752
                     03E3   753 NO_INIT:                                 ; NOT OK TO INITIALIZE
```

```
                50    D4  03E3    754          CLRL    R0                      ; SET FAILURE
                05  03E5    755          RSB                             ; RETURN
                    03E6    756
```

```
                     03E6   758          .SBTTL  INITIALIZE THE DATAPAGE
                     03E6   759 ;++
                     03E6   760 ;
                     03E6   761 ; INIT_DATAPAGE - INITIALIZE THE DATAPAGE
                     03E6   762 ;
                     03E6   763 ; THIS ROUTINE IS CALLED TO INTIALIZE THE SHARED MEMORY DATAPAGE
                     03E6   764 ; FIELDS AND ALLOCATE THE OTHER DATA STRUCTURES.
                     03E6   765 ;
                     03E6   766 ; INPUTS:
                     03E6   767 ;
                     03E6   768 ;       R4 = ADDR OF NEXUS CSR
                     03E6   769 ;       R5 = ADDR OF SHB
                     03E6   770 ;       R6 = ADDR OF DATAPAGE (SHD)
                     03E6   771 ;       SHR_VALUES = LIST OF SHARE COMMAND QUALIFIER VALUES
                     03E6   772 ;
                     03E6   773 ; OUTPUTS:
                     03E6   774 ;
                     03E6   775 ;       R0 = SUCCESS OR FAILURE STATUS.
                     03E6   776 ;
                     03E6   777 ;       SHARED MEMORY DATAPAGE IS INITIALIZED.
                     03E6   778 ;
                     03E6   779 ;       DURING INITIALIZATION, THE OTHER DATASTRUCTURES ARE ALLOCATED
                     03E6   780 ;       SO THAT THE APPEAR IN THE FOLLOWING ORDER IN VIRTUAL MEMORY:
                     03E6   781 ;
                     03E6   782 ;               +-----------------------+
                     03E6   783 ;               |       DATAPAGE        |
                     03E6   784 ;               +-----------------------+
                     03E6   785 ;               \     PER PORT PAGES    \
                     03E6   786 ;               +-----------------------+
                     03E6   787 ;               |       PRQ'S           |
                     03E6   788 ;               +-----------------------+
                     03E6   789 ;               |       GSD'S           |
                     03E6   790 ;               +-----------------------+
                     03E6   791 ;               |       MBX'S           |
                     03E6   792 ;               +-----------------------+
                     03E6   793 ;               |       CEF'S           |
                     03E6   794 ;               +-----------------------+
                     03E6   795 ;               |       POOL            |
                     03E6   796 ;               +-----------------------+
                     03E6   797 ;               |       BITMAP          |
                     03E6   798 ;               +-----------------------+
                     03E6   799 ;               |                       |
                     03E6   800 ;               \  GLOBAL SECTION PAGES \
                     03E6   801 ;               |                       |
                     03E6   802 ;               +-----------------------+
                     03E6   803 ;
                     03E6   804 ;       *** NOTE: THE ORDER OF THESE STRUCTURES IS ASSUMED ****
                     03E6   805 ;--
                     03E6   806 INIT_DATAPAGE:                               ; INITIALIZE THE DATAPAGE
          01   90    03E6   807          MOVB    #SHD$M_INITLCK,-             ; CLEAR THE FLAGS BUT KEEP
     009F C6           03E8   808                  SHD$B_FLAGS(R6)              ;  THE LOCK SET
     0000'CF   90      03EB   809          MOVB    SHR_Q_MEMNAME,-              ; SET MEMORY NAME SIZE
       20 A6           03EF   810                  SHD$T_NAME(R6)               ;
          30   BB      03F1   811          PUSHR   #^M<R4,R5>                   ; SAVE MOVC REGISTERS
0004'DF   0000'CF   2C  03F3   812          MOVC5   SHR_Q_MEMNAME,@SHR_Q_MEMNAME+4,- ; SET MEMORY NAME STRING
  21 A6   0F   00       03FA   813                  #0,#15,SHD$T_NAME+T(R6)      ; ZERO-FILLED TO 15 TEXT BYTES
          30   BA      03FE   814          POPR    #^M<R4,R5>                   ; RESTORE MOVC REGISTERS
```

```
                00000000'GF   7D  0400    815          MOVQ    G^EXE$GQ_SYSTIME,-      ; SET INITIALIZATION TIME
                      30 A6       0406    816                  SHD$Q_INITTIME(R6)      ;
                         50   D4  0408    817          CLRL    R0                      ;INIT PORT NUMBER
                                  040A    818          ASSUME  <SHD$Q_PRQWRK & ^B111> EQ 0 ; LIST HEADS MUST BE QUADWORD ALIGNED
                  0100 C640   7C  040A    819 10$:     CLRQ    SHD$Q_PRQWRK(R6)[R0]    ;INIT PORT'S REQUEST WORK QUEUE
                F7 50    04   F2  040F    820          AOBLSS  #MPM$C_PORTS,R0,10$     ;INCREMENT PORT NUMBER AND LOOP
                   00A4 C6   B4  0413    821          CLRW    SHD$W_PRQWAIT(R6)       ;INIT PRQ WAIT FLAGS
                   00E8 C6   B4  0417    822          CLRW    SHD$W_RESSUM(R6)        ;INIT RESOURCE REPORT SUMMARY FLAGS
                         50   D4  041B    823          CLRL    R0                      ;SET STARTING RESOURCE NUMBER
                   00C8 C640   B4  041D    824 20$:     CLRW    SHD$W_RESAVAIL(R6)[R0]  ;INIT RESOURCE AVAILABLE FLAGS
                   00A8 C640   B4  0422    825          CLRW    SHD$W_RESWAIT(R6)[R0]   ;INIT RESOURCE WAIT FLAGS
                F2 50    0F   F2  0427    826          AOBLSS  #RSN$_MAX,R0,20$        ;INCREMENT RESOURCE NUMBER AND LOOP
               009C C6    04   90  042B    827          MOVB    #MPM$C_PORTS,SHD$B_PORTS(R6) ; SET NUMBER OF PORTS
                   0022'CF   D0  0430    828          MOVL    SHR_L_START,-           ; SET RELATIVE PFN
                      14 A6       0434    829                  SHD$L_GSPFN(R6)         ;  OF 1ST GLOBAL PAGE
                      14 A6   C1  0436    830          ADDL3   SHD$L_GSPFN(R6),-       ; SET PFN OF 1ST GLOBAL PAGE
            10 A5     002B'CF       0439    831                  SHR_L_MEMPFN,SHB$L_BASGSPFN(R5)
            18 A6     000A'CF   B0  043E    832          MOVW    SHR_W_GBLCNT,SHD$W_GSDMAX(R6) ; SET GLOBAL SECTION DESC COUNT
            1A A6     000C'CF   B0  0444    833          MOVW    SHR_W_MBXCNT,SHD$W_MBXMAX(R6) ; SET MAILBOX COUNT
            1C A6     000E'CF   B0  044A    834          MOVW    SHR_W_CEFCNT,SHD$W_CEFMAX(R6) ; SET COMMON EVT FLAG CLUST COUNT
                  50     15 A5   9A  0450    835          MOVZBL  SHB$B_PORT(R5),R0       ; GET THIS PORT'S PORT NUMBER
         3C A640     0010'CF   B0  0454    836          MOVW    SHR_W_GBLQUO,SHD$W_GSDQUOTA(R6)[R0] ; SET THIS PORT'S GSD QUOTA
         5C A640     0012'CF   B0  045B    837          MOVW    SHR_W_MBXQUO,SHD$W_MBXQUOTA(R6)[R0] ; SET THIS PORT'S MBX QUOTA
         7C A640     0014'CF   B0  0462    838          MOVW    SHR_W_CEFQUO,SHD$W_CEFQUOTA(R6)[R0] ; SET THIS PORT'S CEF QUOTA
            18 A6    3C A640   B1  0469    839          CMPW    SHD$W_GSDQUOTA(R6)[R0],SHD$W_GSDMAX(R6) ; IS QUOTA > TABLE SIZE?
                        06     15  046F    840          BLEQ    30$                     ; BR IF QUOTA IS OK
            3C A640     18 A6   B0  0471    841          MOVW    SHD$W_GSDMAX(R6),SHD$W_GSDQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
            1A A6    5C A640   B1  0477    842 30$:     CMPW    SHD$W_MBXQUOTA(R6)[R0],SHD$W_MBXMAX(R6) ; IS QUOTA > TABLE SIZE?
                        06     15  047D    843          BLEQ    40$                     ; BR IF QUOTA IS OK
            5C A640     1A A6   B0  047F    844          MOVW    SHD$W_MBXMAX(R6),SHD$W_MBXQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
            1C A6    7C A640   B1  0485    845 40$:     CMPW    SHD$W_CEFQUOTA(R6)[R0],SHD$W_CEFMAX(R6) ; IS QUOTA > TABLE SIZE?
                        06     15  048B    846          BLEQ    50$                     ; BR IF QUOTA IS OK
            7C A640     1C A6   B0  048D    847          MOVW    SHD$W_CEFMAX(R6),SHD$W_CEFQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
                                  0493    848 50$:
                                  0493    849 ;
                                  0493    850 ; FILL-IN DATAPAGE RELATIVE ADDRESSES OF OTHER DATA STRUCTURES AND
                                  0493    851 ; AND KEEP A TOTAL OF THE NUMBER OF PAGES NEEDED FOR THE STUCTURES.
                                  0493    852 ;
                  50    01FF 8F   3C  0493    853          MOVZWL  #511,R0                 ; GET SIZE OF PAGE - 1
                  51    0200 8F   3C  0498    854          MOVZWL  #512,R1                 ; GET SIZE OF A PAGE
                        53    01  D0  049D    855          MOVL    #1,R3                   ; INIT RELATIVE PAGE POINTER
                        53    04  C0  04A0    856          ADDL    #MPM$C_PORTS,R3         ; RESERVE PER PORT PAGES
               00F0 C6  53    09  78  04A3    857          ASHL    #VA$V_VPN,R3,-          ; SET RELATIVE ADDR OF PRQ FREE LIST
                                  04A9    858                  SHD$Q_PRQ(R6)           ;
                 00000040 8F      C5  04A9    859          MULL3   #PRQ$C_MINLENGTH,-      ; COMPUTE NUMBER BYTES FOR PRQ'S
                  52    001E'CF       04AF    860                  SHR_L_PRQCNT,R2         ;
                        52    50  C0  04B3    861          ADDL    R0,R2                   ; ROUND-UP TO A PAGE
                        52    51  C6  04B6    862          DIVL    R1,R2                   ; CONVERT TO PAGES
                        53    52  C0  04B9    863          ADDL    R2,R3                   ; COMPUTE RELATIVE PAGE OF GSD'S
            04 A6       53    09  78  04BC    864          ASHL    #VA$V_VPN,R3,-          ; SET RELATIVE ADDR OF GSD TABLE
                                  04C1    865                  SHD$L_GSDPTR(R6)        ;
                  52    000A'CF   3C  04C1    866          MOVZWL  SHR_W_GBLCNT,R2         ; GET NUMBER OF GSD'S
                   00000074 8F      C1  04C6    867          ADDL3   #GSD$K_SHMGSDLNG,-      ; COMPUTE SIZE OF GSD'S
                  002F'CF    10       04CC    868                  #<MPM$C_PORTS*4>,SHR_L_GSDSIZE ; (LONGWORD REFCNT/PORT)
                  52    002F'CF   C4  04D0    869          MULL    SHR_L_GSDSIZE,R2        ; COMPUTE NUMBER BYTES NEEDED
                        52    50  C0  04D5    870          ADDL    R0,R2                   ; ROUND-UP TO A PAGE
                        52    51  C6  04D8    871          DIVL    R1,R2                   ; CONVERT TO PAGES
```

```
                53   52   C0  04DB   872          ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF MBX'S
             66 53   09   78  04DE   873          ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF MBX TABLE
                                 04E2   874                  SHD$L_MBXPTR(R6)                ;
             52 000C'CF   3C  04E2   875          MOVZWL  SHR_W_MBXCNT,R2                 ; GET NUMBER OF MAILBOXES
                52   30   C4  04E7   876          MULL    #MBX$K_LENGTH,R2                ; COMPUTE NUMBER BYTES NEEDED
                52   50   C0  04EA   877          ADDL    R0,R2                           ; ROUND-UP TO A PAGE
                52   51   C6  04ED   878          DIVL    R1,R2                           ; CONVERT TO PAGES
                53   52   C0  04F0   879          ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF CEF TABLE
          08 A6 53   09   78  04F3   880          ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF CEF TABLE
                                 04F8   881                  SHD$L_CEFPTR(R6)                ;
             52 000E'CF   3C  04F8   882          MOVZWL  SHR_W_CEFCNT,R2                 ; GET NUMBER OF COM EVT FLAG BLOCKS
                   38   C1  04FD   883          ADDL3   #CEB$C_LENGTH,-                 ; COMPUTE SIZE OF SHMCEB
          0033'CF  18        04FF   884                  #<MPM$C_PORTS*6>,SHR_L_CEFSIZE ; (WORD REFCNT+SLAVE VA/PORT)
             52 0033'CF   C4  0503   885          MULL    SHR_L_CEFSIZE,R2                ; COMPUTE NUMBER BYTES NEEDED
                52   50   C0  0508   886          ADDL    R0,R2                           ; ROUND-UP TO A PAGE
                52   51   C6  050B   887          DIVL    R1,R2                           ; CONVERT TO PAGES
                53   52   C0  050E   888          ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF POOL
       00F8 C6  53   09   78  0511   889          ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF POOL
                                 0517   890                  SHD$Q_POOL(R6)                  ;
             0016'CF   C5  0517   891          MULL3   SHR_L_POOLBCNT,-                ; COMPUTE SIZE OF POOL IN BYTES
             52 001A'CF         051B   892                  SHR_L_POOLBSIZ,R2               ;
                52   50   C0  051F   893          ADDL    R0,R2                           ; ROUND-UP TO A PAGE
                52   51   C6  0522   894          DIVL    R1,R2                           ; CONVERT TO PAGES
                53   52   C0  0525   895          ADDL    R2,R3                           ; COMPUTE RELATIVE PAGE OF BITMAP
          0C A6 53   09   78  0528   896          ASHL    #VA$V_VPN,R3,-                  ; SET RELATIVE ADDR OF BITMAP
                                 052D   897                  SHD$L_GSBITMAP(R6)              ;
                                 052D   898 ;
                                 052D   899 ; COMPUTE NUMBER PAGES LEFT FOR GLOBAL SECTIONS AND SIZE OF BITMAP
                                 052D   900 ; TO REPRESENT GLOBAL SECTION PAGES.
                                 052D   901 ;
                   14 A6   C3  052D   902          SUBL3   SHD$L_GSPFN(R6),-               ; GET MEMORY SIZE
             50 0027'CF         0530   903                  SHR_L_MEMSIZE,R0                ;  LESS RESERVED AREA
                50   53   C2  0534   904          SUBL    R3,R0                           ; COMPUTE NUMBER GLOBAL SECTION
                                 0537   905                                                  ; PAGES THAT WILL BE AVAILABLE
                   21   15  0537   906          BLEQ    100$                            ; BRANCH IF THERE ARE NONE
51   50   00000FFF 8F   C1  0539   907          ADDL3   #<512*8>-1,R0,R1                ; ROUND-UP TO NUMBER PER
                                 0541   908                                                  ;  PAGE OF BITMAP
     51   00001000 8F   C6  0541   909          DIVL    #<512*8>,R1                     ; COMPUTE NUMBER PAGES FOR BITMAP
      10 A6   50   51   C3  0548   910          SUBL3   R1,R0,SHD$L_GSPAGCNT(R6) ; SET NUMBER GLOBAL PAGES AVAIL
                                 054D   911                                                  ;  LESS NUMBER BITMAP PAGES
                   0B   15  054D   912          BLEQ    100$                            ; BRANCH IF NONE
                02E2   30  054F   913          BSBW    DATAPAGE_CRC                    ; COMPUTE DATAPAGE CRC
          38 A6   50   D0  0552   914          MOVL    R0,SHD$L_CRC(R6)                ; SET CRC
                50   01   D0  0556   915          MOVL    #1,R0                           ; SET SUCCESS
                     05   05  0559   916          RSB                                     ; RETURN
                                 055A   917
                                 055A   918 100$:
     50   007C804A 8F   D0  055A   919          MOVL    #SYSG$_BADPARAM,R0              ; SET FAILURE
                     05   05  0561   920          RSB                                     ; RETURN
                                 0562   921
```

```
                                0562    923          .SBTTL  MAP THE OTHER DATA STRUCTURES
                                0562    924 ;++
                                0562    925 ;
                                0562    926 ; MAP_STRUCTURES - MAP THE OTHER DATA STRUCTURES
                                0562    927 ;
                                0562    928 ; THIS ROUTINE IS CALLED TO MAP THE OTHER DATA STRUCTURES INTO
                                0562    929 ; SYSTEM VIRTUAL ADDRESS SPACE.
                                0562    930 ;
                                0562    931 ; INPUTS:
                                0562    932 ;
                                0562    933 ;       R4 = ADDR OF CSR NEXUS
                                0562    934 ;       R5 = ADDR OF SHB
                                0562    935 ;       R6 = ADDR OF DATAPAGE (SHD)
                                0562    936 ;
                                0562    937 ; OUTPUTS:
                                0562    938 ;
                                0562    939 ;       R0 = SUCCESS OR FAILURE STATUS.
                                0562    940 ;
                                0562    941 ;       THE OTHER SHARED MEMORY DATA STRUCTURES (POOL, MAILBOXES,
                                0562    942 ;       GLOBAL SECTION DESCRIPTORS, GLOBAL SECTION BITMAP) ARE
                                0562    943 ;       MAPPED IN SYSTEM SPACE.  THE RELATIVE ADDRESSES IN THE
                                0562    944 ;       DATAPAGE CAN NOW BE USED TO ACCESS THE STRUCTURES.
                                0562    945 ;
                                0562    946 ;       NOTE: THE PAGES ARE MAPPED SO THAT THE HIGHEST NUMBERED PFN
                                0562    947 ;       HAS THE LOWEST VIRTUAL ADDRESS.
                                0562    948 ;
                                0562    949 ;--
                                0562    950 MAP_STRUCTURES:                             ; MAP THE OTHER STRUCTURES
                      F7 8F   78    0562    951          ASHL    #-VA$V_VPN,-          ; GET NUMBER OF PAGES TO MAP
                   51   0C A6         0565    952                  SHD$L_GSBITMAP(R6),R1 ; (BITMAP IS LAST STRUCTURE)
                00000000'GF   16    0568    953          JSB     G^IOC$ALLOSPT         ; ALLOCATE SYS PAGE TABLE ENTRIES
                         08 50   E8    056E    954          BLBS    R0,10$                ; BRANCH IF SUCCESS
                50   007C8022 8F   D0    0571    955          MOVL    #SYSG$_SPTFULL,R0     ; SET FAILURE STATUS
                                05    0578    956          RSB                           ; RETURN
                                      0579    957 10$:
                50   56   15   09   EF    0579    958          EXTZV   #VA$V_VPN,#VA$S_VPN,R6,R0 ; GET VPN OF DATAPAGE
                   003B'CF   6340   DE    057E    959          MOVAL   (R3)[R0],SHR_L_SHDPTE ; SAVE ADDR OF DATAPAGE PTE
                      50   6340   D0    0584    960          MOVL    (R3)[R0],R0           ; GET PTE OF DATAPAGE (LAST PAGE)
                50   50   15   00   EF    0588    961          EXTZV   #PTE$V_PFN,#PTE$S_PFN,R0,R0 ; GET PFN OF DATAPAGE
                                      058D    962 20$:                              ; MAPPING LOOP
                               50   D7    058D    963          DECL    R0                    ; DECREMENT PFN
                      B0000000 8F   C9    058F    964          BISL3   #<PTE$C_ERKW!PTE$M_VALID>,- ; SET PTE VALID, KERNEL WRITEABLE
                        6342   50         0595    965                  R0,(R3)[R2]           ;  AND ENTER PFN
                               52   D6    0598    966          INCL    R2                    ; INCREMENT VPN
                            F0 51   F5    059A    967          SOBGTR  R1,20$                ; DECREMENT PAGE COUNT AND LOOP
                                      059D    968
             18 A5   56   0C A6   C1    059D    969          ADDL3   SHD$L_GSBITMAP(R6),R6,- ; SET ADDR OF END OF POOL
                                      05A3    970                  SHB$L_POOLEND(R5)     ;  (BITMAP ASSUMED TO FOLLOW POOL)
                         50   01   D0    05A3    971          MOVL    #1,R0                 ; SET SUCCESS
                                05    05A6    972          RSB                           ; RETURN
                                      05A7    973
```

```
                     05A7   975          .SBTTL  INITIALIZE THE OTHER DATA STRUCTURES
                     05A7   976 ;++
                     05A7   977 ;
                     05A7   978 ; INIT_STRUCTURES - INITIALIZE THE OTHER DATA STRUCTURES
                     05A7   979 ;
                     05A7   980 ; THIS ROUTINE IS CALLED TO INTIALIZE THE OTHER DATA STRUCTURES IN
                     05A7   981 ; THE SHARED MEMORY.
                     05A7   982 ;
                     05A7   983 ; INPUTS:
                     05A7   984 ;
                     05A7   985 ;       R4 = ADDR OF NEXUS CSR
                     05A7   986 ;       R5 = ADDR OF SHB
                     05A7   987 ;       R6 = ADDR OF DATAPAGE (SHD)
                     05A7   988 ;
                     05A7   989 ; OUTPUTS:
                     05A7   990 ;
                     05A7   991 ;       THE OTHER SHARED MEMORY DATA STRUCTURES (POOL, MAILBOXES,
                     05A7   992 ;       GLOBAL SECTION DESCRIPTORS, GLOBAL SECTION BITMAP) ARE
                     05A7   993 ;       INITIALIZED FOR USE.
                     05A7   994 ;--
                     05A7   995 INIT_STRUCTURES:                        ; INITIALIZE THE STRUCTURES
                     05A7   996 ;
                     05A7   997 ; INITIALIZE THE GLOBAL SECTION DESCRIPTOR TABLE
                     05A7   998 ;
          57   56  04 A6   C1  05A7   999          ADDL3   SHD$L_GSDPTR(R6),R6,R7  ; GET ADDR OF 1ST GSD
               58  18 A6   3C  05AC  1000          MOVZWL  SHD$W_GSDMAX(R6),R8     ; GET COUNT OF GSD'S
                               05B0  1001 10$:                                    ; INIT LOOP
                               05B0  1002          ASSUME GSD$L_GSDBL EQ <GSD$L_GSDFL + 4>
                         67    7C  05B0  1003          CLRQ    GSD$L_GSDFL(R7)     ; INIT FORWARD/BACKWARD LINKS
                    009C C6    90  05B2  1004          MOVB    SHD$B_PORTS(R6),-   ; SET # OF PORTS
                      51 A7        05B6  1005                  GSD$B_PROCCNT(R7)   ;
                         29    9B  05B8  1006          MOVZBW  #DYN$C_SHMGSD,-     ; SET STRUCTURE TYPE AND
                      0A A7        05BA  1007                  GSD$B_TYPE(R7)      ;   CLEAR UNUSED BYTE
                    002F'CF    B0  05BC  1008          MOVW    SHR_L_GSDSIZE,-     ; SET STRUCTURE SIZE
                      08 A7        05C0  1009                  GSD$W_SIZE(R7)      ;
                      50 A7    94  05C2  1010          CLRB    GSD$B_LOCK(R7)      ; CLEAR LOCK
                      53 A7    94  05C5  1011          CLRB    GSD$B_DELETPORT(R7) ; CLEAR ANY DELETES PENDING
                   50   51 A7    9A  05C8  1012          MOVZBL  GSD$B_PROCCNT(R7),R0 ; GET # OF PROCESSOR REF CNTS
                   51   74 A7    DE  05CC  1013          MOVAL   GSD$L_PTECNT1(R7),R1 ; GET ADR OF FIRST REF CNT
                         81    D4  05D0  1014 20$:     CLRL    (R1)+               ; CLEAR ONE REF COUNT
                         FB 50   F5  05D2  1015          SOBGTR  R0,20$              ; REPEAT FOR EACH PROCESSOR REF COUNT
                   57  002F'CF    C0  05D5  1016          ADDL    SHR_L_GSDSIZE,R7    ; INCREMENT GSD POINTER
                      D3 58    F5  05DA  1017          SOBGTR  R8,10$              ; DECREMENT GSD COUNTER AND LOOP
                               05DD  1018 ;
                               05DD  1019 ; INITIALIZE THE GLOBAL SECTION BITMAP BY PERFORMING A GROSS READ/WRITE
                               05DD  1020 ; TEST ON EACH PAGE.  IF IT PASSES THE TEST, IT IS MARKED AVAILABLE
                               05DD  1021 ; IN THE BITMAP.  IF IT DOESN'T PASS THE  TEST, IT IS MARKED UNAVAILABLE.
                               05DD  1022 ;
                               05DD  1023 ; *** NOTE: TO MAP THE PAGES, THE DATAPAGE IS UNMAPPED ***
                               05DD  1024 ;
                               05DD  1025 INIT_BITMAP:                            ; INITIALIZE THE BITMAP
                         54    DD  05DD  1026          PUSHL   R4                  ; SAVE REGISTER
          57   56  0C A6    C1  05DF  1027          ADDL3   SHD$L_GSBITMAP(R6),R6,R7 ; GET ADDR OF BITMAP
                         3F    BB  05E4  1028          PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; SAVE REGISTERS DESTROYED BY MOVC
67  0200 8F    00   66    00    2C  05E6  1029          MOVC5   #0,(R6),#0,#512,(R7)    ; ZERO-FILL THE BITMAP PAGE
                         3F    BA  05EE  1030          POPR    #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE REGISTERS CLOBBERED BY MOVC
               58  003B'CF    D0  05F0  1031          MOVL    SHR_L_SHDPTE,R8     ; GET ADDR OF DATAPAGE PTE
```

```
                        68   DD  05F5  1032          PUSHL   (R8)                            ; SAVE DATAPAGE PTE
                    59  10 A6   D0  05F7  1033          MOVL    SHD$L_GSPAGCNT(R6),R9       ; GET NUMBER OF GLOBAL PAGES
                    5A  10 A5   D0  05FB  1034          MOVL    SHB$L_BASGSPFN(R5),R10      ; GET PFN OF 1ST GLOBAL PAGE
                        5B   D4  05FF  1035          CLRL    R11                             ; INI. CURRENT RELATIVE PAGE NUMBER
                                 0601  1036 10$:
68  15  00  5A   F0  0601  1037          INSV    R10,#PTE$V_PFN,#PTE$S_PFN,(R8) ; MAP THE PAGE TO TEST
        53  56   D0  0606  1038          MOVL    R6,R3                           ; GET A COPY OF VIRTUAL ADDRESS OF PAGE
    54  5A  09   78  0609  1039          ASHL    #VA$V_VPN,R10,R4                ; COMPUTE PHYSICAL ADDRESS OF PAGE
                             060D  1040          INVALID R3                      ; INVALIDATE VIRTUAL ADDRESS TRANSLATION
            F9ED'   30  0610  1041          BSBW    IOGEN$TEST_MEM          ; TEST THE PAGE
          06 50     E9  0613  1042          BLBC    R0,20$                  ; BR IF BAD PAGE
    06 67   5B      E2  0616  1043          BBSS    R11,(R7),30$            ; SET PAGE OK
            04      11  061A  1044          BRB     30$                     ;
                        061C  1045 20$:
    00 67   5B      E5  061C  1046          BBCC    R11,(R7),30$            ; SET PAGE BAD
                        0620  1047 30$:
            5A      D6  0620  1048          INCL    R10                     ; INCREMENT PFN
    DB 5B   59      F2  0622  1049          AOBLSS  R9,R11,10$              ; INCREMENT CURRENT PAGE NUMBER AND LOOP
            68 8ED0     0626  1050          POPL    (R8)                    ; RESTORE DATAPAGE PTE (REMAP)
                        0629  1051          INVALID R6                      ; INVALIDATE VIRTUAL ADDRESS TRANSLATION
            54 8ED0     062C  1052          POPL    R4                      ; RESTORE REGISTER
 00000000'GF    16      062F  1053          JSB     G^MA$INITIAL            ; CLEAR ANY PORT ERRORS
                        0635  1054 ;
                        0635  1055 ; INITIALIZE THE POOL
                        0635  1056 ;
                        0635  1057 INIT_POOL:                               ; INITIALIZE THE POOL
50  56  00F8 C6  C1  0635  1058          ADDL3   SHD$Q_POOL(R6),R6,R0    ; GET ADDR OF FIRST B OCK
        00F8 C6  7C  063B  1059          CLRQ    SHD$Q_POOL(R6)          ; SET QUEUE EMPTY
    51  0016'CF  D0  063F  1060          MOVL    SHR_L_POOLBCNT,R1       ; GET NUMBER OF BLOCKS
    52  001A'CF  D0  0644  1061          MOVL    SHR_L_POOLBSIZ,R2       ; GET BLOCK SIZE
                        0649  1062 10$:
    00F8 C6  60  5C  0649  1063          INSQHI  (R0),SHD$Q_POOL(R6)     ; INSERT BLOCK IN LIST
      08 A0  52  B0  064E  1064          MOVW    R2,ACB$W_SIZE(R0)       ; SET SIZE OF BLOCK IN BLOCK
         50  52  C0  0652  1065          ADDL    R2,R0                   ; INCREMENT BLOCK POINTER
           F1 51  F5  0655  1066          SOBGTR  R1,10$                  ; DECREMENT BLOCK COUNT AND LOOP
                        0658  1067 ;
                        0658  1068 ; INTIALIZE THE FREE INTER-PROCESSOR REQUEST BLOCK QUEUE
                        0658  1069 ;
                        0658  1070 INIT_PRQ:                                ; INITIALIZE FREE PRQ QUEUE
50  56  00F0 C6  C1  0658  1071          ADDL3   SHD$Q_PRQ(R6),R6,R0     ; GET ADDR OF FIRST BLOCK
        00F0 C6  7C  065E  1072          CLRQ    SHD$Q_PRQ(R6)           ; SET QUEUE EMPTY
    51  001E'CF  D0  0662  1073          MOVL    SHR_L_PRQCNT,R1         ; GET NUMBER OF BLOCKS
                        0667  1074 10$:
    00F0 C6  60  5D  0667  1075          INSQTI  (R0),SHD$Q_PRQ(R6)      ; INSERT BLOCK IN LIST
 50  00000040 8F  C0  066C  1076          ADDL    #PRQ$C_MINLENGTH,R0     ; INCREMENT BLOCK POINTER
           F1 51  F5  0673  1077          SOBGTR  R1,10$                  ; DECREMENT BLOCK COUNT AND LOOP
                        0676  1078
                        0676  1079 ;
                        0676  1080 ; INITIALIZE THE MAILBOX TABLE
                        0676  1081 ;
                        0676  1082 INIT_MAILBOXES:                          ; INITIALIZE THE MAILBOXES
    57  56  66  C1  0676  1083          ADDL3   SHD$L_MBXPTR(R6),R6,R7  ; GET ADDR OF 1ST MAILBOX
      58  1A A6  3C  067A  1084          MOVZWL  SHD$W_MBXMAX(R6),R8     ; GET NUMBER TO INIT
            50  D4  067E  1085          CLRL    R0                      ; INIT INITIALIZED COUNT
                        0680  1086 10$:
          08 A7  94  0680  1087          CLRB    MBX$B_FLAGS(R7)         ; CLEAR ALL FLAGS
 0A A7  50   01  A1  0683  1088          ADDW3   #1,R0,MBX$W_UNIT(R7)    ; SET UNIT NUMBER
```

```
                     0688  1089                                          ; (FROM 1 TO N, AS 0 IS RESERVED)
          57   30    C0  0688  1090          ADDL    #MBX$K_LENGTH,R7      ; INCREMENT MAILBOX POINTER
      F1  50   58    F2  068B  1091          AOBLSS  R8,R0,TO$             ; INCREMENT COUNT AND LOOP
                         068F  1092
                         068F  1093 ;
                         068F  1094 ; INITIALIZE THE COMMON EVENT FLAG TABLE
                         068F  1095 ;
                         068F  1096 INIT_CEF:
57   56    08 A6    C1  068F  1097          ADDL3   SHD$L_CEFPTR(R6),R6,R7 ; GET ADR OF 1ST SHMCEB IN TABLE
     58    1C A6    3C  0694  1098          MOVZWL  SHD$W_CEFMAX(R6),R8   ; GET NUMBER OF ENTRIES TO INIT
                         0698  1099 10$:
                         0698  1100          ASSUME CEB$L_CEBBL EQ <CEB$L_CEBFL+4>
               67    7C  0698  1101          CLRQ    CEB$L_CEBFL(R7)       ; INIT FLAGS
1D A7     009C C6    90  069A  1102          MOVB    SHD$B_PORTS(R6),CEB$B_PROCCNT(R7) ; SET # OF PROCESSORS
   0A A7     2E    9B  069F  1103          MOVZBW  #DYN$C_SHMCEB,CEB$B_TYPE(R7) ; SET TYPE OF DATA STRUCTURE
08 A7   0033'CF    B0  06A4  1104          MOVW    SHR_L_CEFSIZE,CEB$W_SIZE(R7) ; SET SIZE OF SHMCEB
     1C A7    94  06AA  1105          CLRB    CEB$B_LOCK(R7)        ; CLEAR OWNER OF CEB LOCK
     1F A7    94  06AD  1106          CLRB    CEB$B_DELETPORT(R7)   ; CLEAR DELETOR OF CEB
   50   1D A7    9A  06B0  1107          MOVZBL  CEB$B_PROCCNT(R7),R0  ; GET # OF PROCESSORS MAX
 51   38 A740    DE  06B4  1108          MOVAL   CEB$L_VASLAVE1(R7)[R0],R1 ; GET ADR OF FIRST PROC REF COUNT
   52   38 A7    DE  06B9  1109          MOVAL   CEB$L_VASLAVE1(R7),R2 ; GET ADR OF FIRST SLAVE CEB VA
          82    D4  06BD  1110 20$:     CLRL    (R2)+                 ; CLEAR THE VA OF SLAVE CEB FOR PROC
          81    B4  06BF  1111          CLRW    (R1)+                 ; CLEAR REF COUNT FOR THIS PROCESSOR
        F9 50    F5  06C1  1112          SOBGTR  R0,20$                ; REPEAT FOR EACH PROCESSOR
 57   0033'CF    C0  06C4  1113          ADDL    SHR_L_CEFSIZE,R7      ; GET NEXT SHMCEB IN TABLE
        CC 58    F5  06C9  1114          SOBGTR  R8,TO$                ; INIT EACH SHMCEB IN TABLE
                         06CC  1115
     50   01    D0  06CC  1116          MOVL    #1,R0                 ; SET SUCCESS
                    05  06CF  1117          RSB
                         06D0  1118
```

```
                     06D0  1120          .SBTTL  CONNECT TO OTHER DATA STRUCTURES
                     06D0  1121 ;++
                     06D0  1122 ;
                     06D0  1123 ; CONNECT_MEM - CONNECT TO OTHER SHARED MEMORY DATA STRUCTURES
                     06D0  1124 ;
                     06D0  1125 ; THIS ROUTINE IS CALLED TO JUST CONNECT THIS PORT TO AN ALREADY
                     06D0  1126 ; INTIALIZED SHARED MEMORY.
                     06D0  1127 ;
                     06D0  1128 ; INPUTS:
                     06D0  1129 ;
                     06D0  1130 ;       R4 = ADDR OF NEXUS CSR
                     06D0  1131 ;       R5 = ADDR OF SHB
                     06D0  1132 ;       R6 = ADDR OF DATAPAGE (SHD)
                     06D0  1133 ;
                     06D0  1134 ; OUTPUTS:
                     06D0  1135 ;
                     06D0  1136 ;       R0 = SUCCESS OR FAILURE STATUS.
                     06D0  1137 ;
                     06D0  1138 ;       THE DATAPAGE IS FIRST TESTED TO BE SURE THAT IT IS INITIALIZED.
                     06D0  1139 ;       IF IT HAS BEEN INITIALIZED, THEN THE EXISTING DATA STRUCTURES
                     06D0  1140 ;       (MAILBOXES, GLOBAL SECTION DESCRIPTORS) ARE SCANNED FOR
                     06D0  :141 ;       ONES THAT ARE MARKED FOR DELETE.  IF THE STRUCTURE HAD
                     06D0  1142 ;       REFERENCES ONLY FROM THIS PORT, THEN THE STRUCTURE IS DELETED.
                     06D0  1143 ;
                     06D0  1144 ;--
                     06D0  1145 CONNECT_MEM:                                 ; CONNECT DATA STRUCTURES
           0161   30 06D0  1146          BSBW    DATAPAGE_CRC               ; COMPUTE DATAPAGE CRC
      38 A6    50 D1 06D3  1147          CMPL    R0,SHD$L_CRC(R6)           ; CRC COMPARE?
               08 13 06D7  1148          BEQL    10$                        ; BRANCH IF YES
   50  007C8052 8F D0 06D9  1149          MOVL    #SYSG$_BADCHKSUM,R0        ; SET FAILURE
               05 06E0  1150          RSB                                ; RETURN
                  06E1  1151 10$:
        50   20 A6 9A 06E1  1152          MOVZBL  SHD$T_NAME(R6),R0          ; GET SIZE OF MEMORY NAME
        21 A6    50 2D 06E5  1153          CMPC5   R0,SHD$T_NAME+1(R6),-      ; IS NAME THE ONE SPECIFIED?
0004'DF  0000'CF    00 06E9  1154                  #0,SHR_Q_MEMNAME,@SHR_Q_MEMNAME+4
               08 13 06F0  1155          BEQLU   20$                        ; BRANCH IF YES
   50  007C805A 8F D0 06F2  1156          MOVL    #SYSG$_INCMEMNAM,R0        ; SET FAILURE
               05 06F9  1157          RSB                                ; RETURN
                  06FA  1158
                  06FA  1159 20$:
           14 A6 C1 06FA  1160          ADDL3   SHD$L_GSPFN(R6),-          ; SET PFN OF 1ST GLOBAL PAGE
  10 A5  002B'CF    06FD  1161                  SHR_L_MEMPFN,SHB$L_BASGSPFN(R5)
  000C'CF    1A A6 B0 0702  1162          MOVW    SHD$W_MBXMAX(R6),SHR_W_MBXCNT ; SAVE NUMBER OF MAILBOXES
        50   15 A5 9A 0708  1163          MOVZBL  SHB$B_PORT(R5),R0          ; GET THIS PORT'S PORT NUMBER
3C A640    0010'CF B0 070C  1164          MOVW    SHR_W_GBLQUO,SHD$W_GSDQUOTA(R6)[R0] ; SET THIS PORT'S GSD QUOTA
5C A640    0012'CF B0 0713  1165          MOVW    SHR_W_MBXQUO,SHD$W_MBXQUOTA(R6)[R0] ; SET THIS PORT'S MBX QUOTA
7C A640    0014'CF B0 071A  1166          MOVW    SHR_W_CEFQUO,SHD$W_CEFQUOTA(R6)[R0] ; SET THIS PORT'S CEF QUOTA
  18 A6    3C A640 B1 0721  1167          CMPW    SHD$W_GSDQUOTA(R6)[R0],SHD$W_GSDMAX(R6) ; IS QUOTA > TABLE SIZE?
               06 15 0727  1168          BLEQ    30$                        ; BR IF QUOTA IS OK
  3C A640    18 A6 B0 0729  1169          MOVW    SHD$W_GSDMAX(R6),SHD$W_GSDQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
  1A A6    5C A640 B1 072F  1170 30$:     CMPW    SHD$W_MBXQUOTA(R6)[R0],SHD$W_MBXMAX(R6) ; IS QUOTA > TABLE SIZE?
               06 15 0735  1171          BLEQ    40$                        ; BR IF QUOTA IS OK
  5C A640    1A A6 B0 0737  1172          MOVW    SHD$W_MBXMAX(R6),SHD$W_MBXQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
  1C A6    7C A640 B1 073D  1173 40$:     CMPW    SHD$W_CEFQUOTA(R6)[R0],SHD$W_CEFMAX(R6) ; IS QUOTA > TABLE SIZE?
               06 15 0743  1174          BLEQ    50$                        ; BR IF QUOTA IS OK
  7C A640    1C A6 B0 0745  1175          MOVW    SHD$W_CEFMAX(R6),SHD$W_CEFQUOTA(R6)[R0] ; MINIMIZE QUO W/TBL SIZ
                  074B  1176 50$:
```

```
                                074B  1177 ;
                                074B  1178 ; RE-INITIALIZE THE REFERENCE COUNTS FOR THIS PORT IN THE GSD TABLE.
                                074B  1179 ; ALSO, IF THIS PORT CREATED ANY OF THE SECTIONS, SET THE CREATOR TO -1
                                074B  1180 ; TO PROHIBIT USE OF NON-EXISTANT SECTION TABLE.  THIS DOES NOT ATTEMPT
                                074B  1181 ; TO RELEASE ANY GLOBAL SECTIONS NOT IN USE BY OTHER PORTS OR GLOBAL SECTIONS
                                074B  1182 ; WHICH WERE ONLY PARTIALLY CREATED BY THIS PORT.
                                074B  1183 ;
                         0F BB  074B  1184          PUSHR   #^M<R0,R1,R2,R3>          ; SAVE REGISTERS
50  56  04 A6 C1          074D  1185          ADDL3   SHD$L_GSDPTR(R6),R6,R0    ; GET ADR OF FIRST GSD IN TABLE
    51  18 A6 3C          0752  1186          MOVZWL  SHD$W_GSDMAX(R6),R1       ; GET COUNT OF GSD'S IN TABLE
                23 15      0756  1187          BLEQ    190$                      ; BR IF NO TABLE TO INIT
    52  15 A5 9A          0758  1188          MOVZBL  SHB$B_PORT(R5),R2         ; GET PORT # FOR THIS PROCESSOR
    53  08 A0 3C          075C  1189          MOVZWL  GSD$W_SIZE(R0),R3         ; GET SIZE OF ONE GSD IN BYTES
       74 A042 D4         0760  1190 110$:    CLRL    GSD$L_PTECNT1(R0)[R2]     ; INITIALIZE THE REF CNT FOR THIS PORT
 0D 60    00 E1          0764  1191          BBC     #GSD$V_VALID,GSD$L_GSDFL(R0),120$ ; BR IF SECTION NOT IN USE
 52 A0    52 91          0768  1192          CMPB    R2,GSD$B_CREATPRT(R0)     ; DID THIS PORT CREATE THE SECTION?
             07 12       076C  1193          BNEQ    120$                      ; BR IF IT IS NOT THE CREATOR
 52 A0    00 92          076E  1194          MCOMB   #0,GSD$B_CREATPRT(R0)     ; MAKE THIS NOT THE CREATOR
       16 A0 B4          0772  1195          CLRW    GSD$W_GSTX(R0)            ; SET NO SECTION TABLE ENTRY
    50  53 C0            0775  1196 120$:    ADDL2   R3,R0                     ; GET ADR OF NEXT GSD
      E5 51 F5           0778  1197          SOBGTR  R1,110$                   ; REPEAT FOR EACH GSD IN TABLE
         0F BA           077B  1198 190$:    POPR    #^M<R0,R1,R2,R3>          ; RESTORE REGISTERS
                         077D  1199
                         077D  1200 ;
                         077D  1201 ; RE-INITIALIZE THE REFERENCE FLAGS FOR THIS PORT IN THE MAILBOXES.
                         077D  1202 ; THIS DOES NOT RELEASE ANY MAILBOXES THAT ARE NOT IN USE BY OTHER PORTS.
                         077D  1203 ;
                         077D  1204 200$:
                         077D  1205          LOCK    #SHD$V_MBXLCK,SHD$B_FLAGS(R6) ; LOCK SHM MAILBOX TABLE
 50  56  66 C1             079B  1206          ADDL3   SHD$L_MBXPTR(R6),R6,R0    ; GET ADDR OF FIRST MAILBOX
   51  1A A6 3C            079F  1207          MOVZWL  SHD$W_MBXMAX(R6),R1       ; GET COUNT OF MAILBOXES IN TABLE
   52  15 A5 9A            07A3  1208          MOVZBL  SHB$B_PORT(R5),R2         ; GET PORT # FOR THIS PROCESSOR
                           07A7  1209 210$:
00 0C A0 52 E7             07A7  1210          BBCCI   R2,MBX$W_REF(R0),220$     ; CLEAR PORT FLAG
                           07AC  1211 220$:
   09 A0 52 91             07AC  1212          CMPB    R2,MBX$B_CREATPORT(R0)    ; IS THIS MBX OWNED BY THIS PORT?
         04 12             07B0  1213          BNEQ    230$                      ; BR IF OWNED BY SOME OTHER PORT
      5C A642 B7           07B2  1214          DECW    SHD$W_MBXQUOTA(R6)[R2]    ; SUBTRACT ONE FOR THIS MBX OWNERSHIP
                           07B6  1215 230$:
      50  30 C0            07B6  1216          ADDL    #MBX$K_LENGTH,R0          ; INCREMENT MAILBOX POINTER
        EB 51 F5           07B9  1217          SOBGTR  R1,210$                   ; DECREMENT COUNT AND LOOP
                           07BC  1218          UNLOCK  #SHD$V_MBXLCK,SHD$B_FLAGS(R6) ; UNLOCK SHM MAILBOX TABLE
                           07C5  1219 ;
                           07C5  1220 ; RE-INITIALIZE THE REFERENCE COUNTS FOR THIS PORT IN THE CEF TABLE.
                           07C5  1221 ; THIS RELEASES ANY TEMPORARY COMMON EVENT FLAG CLUSTERS THAT ARE NOT
                           07C5  1222 ; IN USE BY OTHER PORTS.
                           07C5  1223 ;
                         3F BB 07C5  1224 300$:    PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ;SAVE REGISTERS
50  56  08 A6 C1           07C7  1225          ADDL3   SHD$L_CEFPTR(R6),R6,R0    ; GET ADDR OF FIRST CEB IN TABLE
    51  1C A6 3C           07CC  1226          MOVZWL  SHD$W_CEFMAX(R6),R1       ; GET COUNT OF CEB'S IN TABLE
             5C 15          07D0  1227          BLEQ    390$                      ; BR IF NO CEF TABLE
    52  15 A5 9A           07D2  1228          MOVZBL  SHB$B_PORT(R5),R2         ; GET PORT # FOR THIS PROCESSOR
    53  08 A0 3C           07D6  1229          MOVZWL  CEB$W_SIZE(R0),R3         ; GET SIZE OF ONE CEB IN BYTES
    54  1D A0 9A           07DA  1230          MOVZBL  CEB$B_PROCCNT(R0),R4      ; GET # OF PROCESSOR PORTS ALLOWED
 54 54    02 78             07DE  1231          ASHL    #2,R4,R4                  ; GET # OF BYTES OF SLAVE VA'S
       38 A042 D4           07E2  1232 310$:    CLRL    CEB$L_VASLAVE1(R0)[R2]    ; INDICATE NO SLAVE ENTRY FOR THIS PORT
 55  50  54 C1             07E6  1233          ADDL3   R4,R0,R5                  ; COMPUTE ADR OF CEB PLUS SLAVEVA BYTES
```

```
                50    DD  07EA  1234          PUSHL   R0                      ; REMEMBER CEF ADDRESS
                56    DD  07EC  1235          PUSHL   R6                      ; REMEMBER SHD ADDRESS
           56   50    D0  07EE  1236          MOVL    R0,R6                   ; SET CEF ADDRESS
 00000000'GF    16    07F1  1237          JSB     G^EXE$CEBREFLCK         ; ACQUIRE REFCNT LOCK FOR THIS CEF
     38 A542    B4    07F7  1238          CLRW    CEB$L_VASLAVE1(R5)[R2]  ; CLEAR REFERENCE COUNT FOR PORT
           55   6E    D0  07FB  1239          MOVL    (SP),R5                 ; GET ADR OF SHD
  0E 66    01    E0  07FE  1240          BBS     #CEB$V_LOCKED,CEB$L_CEBFL(R6),320$ ; BR IF LOCKED
  0A 66    00    E1  0802  1241          BBC     #CEB$V_VALID,CEB$L_CEBFL(R6),320$  ; BR IF NOT VALID
  1E A6    52    91  0806  1242          CMPB    R2,CEB$B_CREATPORT(R6)  ; IS THIS CEF OWNED BY THIS PORT?
           04    12  080A  1243          BNEQ    320$                    ; BR IF OWNED BY SOME OTHER PORT
     7C A542    B7  080C  1244          DECW    SHD$W_CEFQUOTA(R5)[R2]  ; SUBTRACT ONE FOR THIS CEF OWNERSHIP
                    0810  1245 320$:
        0F 50    E9  0810  1246          BLBC    R0,340$                 ; BR IF UNABLE TO ACQUIRE REFCNT LOCK
  00 66    02    E7  0813  1247          BBCCI   #CEB$V_REFCNTLCK,CEB$L_CEBFL(R6),330$ ; RELEASE REFCNT LOCK
06 0B A6    01    E0  0817  1248 330$:   BBS     #CEB$V_PERM,CEB$B_STS(R6),340$ ; BR IF PERMANENT CEF
 00000000'GF    16    081C  1249          JSB     G^EXE$SHMCEBDEL         ; RELEASE CEF IF NOT IN USE BY OTHERS
           56 8ED0    0822  1250 340$:   POPL    R6                      ; RESTORE SHD ADDRESS
           50 8ED0    0825  1251          POPL    R0                      ; RESTORE CEF ADDRESS
     50    53    C0  0828  1252          ADDL2   R3,R0                   ; GET ADR OF NEXT CEB IN TABLE
        B4 51    F5  082B  1253          SOBGTR  R1,310$                 ; LOOP TO INIT NEXT CEB
           3F    BA  082E  1254 390$:   POPR    #^M<R0,R1,R2,R3,R4,R5>  ; RESTORE REGISTERS
                    0830  1255
     50    01    D0  0830  1256          MOVL    #1,R0                   ; SET SUCCESS
                05    0833  1257          RSB                             ; RETURN
                    0834  1258
```

```
                     0834  1260          .SBTTL  COMPUTE DATPAGE CRC
                     0834  1261 ;++
                     0834  1262 ;
                     0834  1263 ; DATAPAGE_CRC - COMPUTE DATPAGE CONTSTANT FIELD'S CRC
                     0834  1264 ;
                     0834  1265 ; THIS ROUTINE IS CALLED TO COMPUTE THE CRC OF THE CONSTANT FIELDS
                     0834  1266 ; IN THE SHARED MEMORY DATAPAGE.  THE CRC IS USED TO DETERMINE IF
                     0834  1267 ; THE DATAPAGE IS INTACT.
                     0834  1268 ;
                     0834  1269 ; INPUTS:
                     0834  1270 ;
                     0834  1271 ;       R6 = ADDR OF DATAPAGE
                     0834  1272 ;
                     0834  1273 ; OUTPUTS:
                     0834  1274 ;
                     0834  1275 ;       R0 = CRC OF DATAPAGE CONTSTANT FIELDS
                     0834  1276 ;--
                     0834  1277 DATAPAGE_CRC:                            ; COMPUTE CRC OF DATAPAGE
                     0834  1278          ASSUME  SHD$L_MBXPTR EQ 0       ; ASSUME MBX POINTER IS FIRST
00   F7D2 CF    08   0834  1279          CRC     AUTODIN,#0,-            ; COMPUTE CRC OF DATAPAGE
     66   38         0839  1280                  #<SHD$L_CRC-SHD$L_MBXPTR>,SHD$L_MBXPTR(R6)
                05   083B  1281          RSB                             ; RETURN
```

```
                                        083C  1283          .SBTTL  LOAD SHARED MEMORY MAILBOX DRIVER
                                        083C  1284 ;++
                                        083C  1285 ;
                                        083C  1286 ; LOADMBDRIVER - LOAD SHARED MEMORY MAILBOX DRIVER
                                        083C  1287 ;
                                        083C  1288 ; THIS ROUTINE IS CALLED TO LOAD THE MAILBOX DRIVER AND CONNECT A TEMPLATE
                                        083C  1289 ; MAILBOX UCB TO THE SHARED MEMORY.  THE TEMPLATE IS USED TO CREATE THE USER
                                        083C  1290 ; GENERATED MAILBOX UCB'S BY THE $CREMBX SYSTEM SERVICE.
                                        083C  1291 ;
                                        083C  1292 ; INPUTS:
                                        083C  1293 ;
                                        083C  1294 ;       SHR_L_ADP = ADDRESS OF ADAPTER CONTROL BLOCK
                                        083C  1295 ;
                                        083C  1296 ; OUTPUTS:
                                        083C  1297 ;
                                        083C  1298 ;       SHARED MEMORY MAILBOX DRIVER LOADED, DDB, UCB 0, IDB, AND CRB CREATED
                                        083C  1299 ;       AND CONNECTED TO I/O DATABASE.
                                        083C  1300 ;--
                                        083C  1301 LOADMBDRIVER:                            ; LOAD MAILBOX DRIVER
                                007C    083C  1302          .WORD   ^M<R2,R3,R4,R5,R6>     ;
                                  5E  D8 AE   9E  083E  1303          MOVAB   -ACF$K_LENGTH(SP),SP    ; ALLOCATE CONFIG CONTROL BLOCK
                                     56  5E   D0  0842  1304          MOVL    SP,R6                   ; GET ADDRESS OF BLOCK
                                66   003F'CF   D0  0845  1305          MOVL    SHR_L_ADP,ACF$L_ADAPTER(R6) ; SET ADDRESS OF ADP
                                      0A A6   94  084A  1306          CLRB    ACF$B_AUNIT(R6)         ; SET UNIT #0
                                      0B A6   94  084D  1307          CLRB    ACF$B_AFLAG(R6)         ; SET NO FLAGS
                                      0C A6   D4  0850  1308          CLRL    ACF$L_CONTRLREG(R6)     ; SET NO CONTROL REGISTER ADDR
                                   10 A6   04   B0  0853  1309          MOVW    #PRQ$C_MAILBOX*4,ACF$W_CVECTOR(R6) ; SET MAILBOX VECTOR NUMBER
                                      12 A6   B4  0857  1310          CLRW    ACF$W_CUNIT(R6)         ; SET UNIT #0
                                   1E A6   01   90  085A  1311          MOVB    #1,ACF$B_CNUMVEC(R6)    ; SET ONE VECTOR
                                14 A6   0043'CF   9E  085E  1312          MOVAB   SHR_T_MBDEVNAME,ACF$L_DEVNAME(R6) ; SET ADDRESS OF DEVICE NAME
                0046'CF         0008'CF   42  8F   81  0864  1313          ADDB3   #^A7B7,SHR_W_UNIT,SHR_T_MBDEVNAME+3 ; COMPUTE ''CONTROLLER'' NAME
                                18 A6   F78F CF   9E  086D  1314          MOVAB   SHR_T_MBDRVNAME,ACF$L_DRVNAME(R6) ; SET ADDRESS OF DRIVER NAME
                                1C A6   000C'CF   B0  0873  1315          MOVW    SHR_W_MBXCNT,ACF$W_MAXUNITS(R6) ; SET MAXIMUM NUMBER UNITS
                                   0000'CF   66   FA  0879  1316          CALLG   (R6),IOGEN$LOADER       ; LOAD THE DRIVER AND CONNECT UCB0
                                             04   087E  1317          RET                             ;
                                                  087F  1318
```

```
                                087F  1320          .SBTTL  SHOW THE DATA STRUCTURES
                                087F  1321 ;++
                                087F  1322 ;
                                087F  1323 ; SHOW_STRUCT - SHOW THE DATA STRUCTURES
                                087F  1324 ;
                                087F  1325 ; CALLING SEQUENCE:
                                087F  1326 ;
                                087F  1327 ;       $CMEXEC_S SHOW_STRUCT
                                087F  1328 ;
                                087F  1329 ; INPUTS:
                                087F  1330 ;
                                087F  1331 ;       SHR_L_DATAPAGE = ADDRESS OF SHARED MEMORY DATAPAGE
                                087F  1332 ;
                                087F  1333 ; OUPUTS:
                                087F  1334 ;
                                087F  1335 ;       USEFUL INFORMATION ABOUT THE DATA STRUCTURES IS DISPLAYED
                                087F  1336 ;       ON SYS$OUTPUT.
                                087F  1337 ;--
                                087F  1338 SHOW_STRUCT:                             ; SHOW THE DATA STRUCTURES
                          0FFC  087F  1339          .WORD   ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11> ; ENTRY MASK
      56   0037'CF    D0    0881  1340          MOVL    SHR_L_DATAPAGE,R6           ; GET ADDR OF DATAPAGE
                                0886  1341          PUT_OUTPUT <DATAPAGE:  !XL>,R6
   57   56    04 A6    C1    08B4  1342          ADD[3   SHD$L_GSDPTR(R6),R6,R7  ; GET ADDR OF GSD TABLE
                                08B9  1343          PUT_OUTPUT <GSD TABLE: !XL>,R7
      57   56    66    C1    08E7  1344          ADD[3   SHD$L_MBXPTR(R6),R6,R7  ; GET ADDR OF MBX TABLE
                                08EB  1345          PUT_OUTPUT <MBX TABLE: !XL>,R7
   57   56    08 A6    C1    0919  1346          ADD[3   SHD$L_CEFPTR(R6),R6,R7  ; GET ADDR OF CEF TABLE
                                091E  1347          PUT_OUTPUT <CEF TABLE: !XL>,R7
   57   56    0C A6    C1    094C  1348          ADD[3   SHD$L_GSBITMAP(R6),R6,R7 ; GET ADDR OF BITMAP
                                0951  1349          PUT_OUTPUT <BITMAP:    !XL>,R7
57   56    00F0 C6    C1    097F  1350          ADD[3   SHD$Q_PRQ(R6),R6,R7     ; GET ADDR OF PRQ LIST
                                0985  1351          PUT_OUTPUT <PRQ LIST:  !XL>,R7
                                09B3  1352
                          04    09B3  1353          RET                                 ; RETURN
                                09B4  1354
                                09B4  1355          .END
```

```
$$DESC                  = 000000B5 R      02
$$T2                    = 00000004
ACB$L_KAST              = 00000018
ACB$W_SIZE              = 00000008
ACF$B_AFLAG             = 0000000B
ACF$B_AUNIT             = 0000000A
ACF$B_CNUMVEC           = 0000001E
ACF$K_LENGTH            = 00000028
ACF$L_ADAPTER           = 00000000
ACF$L_CONTRLREG         = 0000000C
ACF$L_DEVNAME           = 00000014
ACF$L_DRVNAME           = 00000018
ACF$W_CUNIT             = 00000012
ACF$W_CVECTOR           = 00000010
ACF$W_MAXUNITS          = 0000001C
ADP$L_CSR               = 00000000
ADP$L_LINK              = 00000004
ADP$L_SHB               = 00000030
ADP$W_ADPTYPE           = 0000000E
ADP$W_TR                = 0000000C
ATS_MPM                   ********    X   03
AUTODIN                   0000000A R      03
BIT...                  = 00000001
CEB$B_CREATPORT         = 0000001E
CEB$B_DELETPORT         = 0000001F
CEB$B_LOCK              = 0000001C
CEB$B_PROCCNT           = 0000001D
CEB$B_STS               = 0000000B
CEB$B_TYPE              = 0000000A
CEB$C_LENGTH            = 00000038
CEB$L_CEBBL             = 00000004
CEB$L_CEBFL             = 00000000
CEB$L_VASLAVE1          = 00000038
CEB$V_LOCKED            = 00000001
CEB$V_PERM              = 00000001
CEB$V_REFCNTLCK         = 00000002
CEB$V_VALID             = 00000000
CEB$W_SIZE              = 00000008
CHECK_INIT                00000395 R      03
CONNECT                   0000024E R      03
CONNECTED                 0000025F R      03
CONNECT_MEM               000006D0 R      03
CREATE_SHB                00000281 R      03
DATAPAGE_CRC              00000834 R      03
DYN$C_SHB               = 0000002A
DYN$C_SHMCEB            = 0000002E
DYN$C_SHMGSD            = 00000029
ERR_EXIT                  0000020E R      03
EXE$CEBREFLCK             ********    X   03
EXE$GL_CONFREGL           ********    X   03
EXE$GL_LOCKRTRY           ********    X   03
EXE$GL_NUMNEXUS           ********    X   03
EXE$GL_RPB                ********    X   03
EXE$GL_SHBLIST            ********    X   03
EXE$GQ_SYSTIME            ********    X   03
EXE$SHMCEBDEL             ********    X   03
EXIT                      0000027D R      03
FIND_UNIT                 000001DE R      03
GEN$SHARE                 00000148 RG     03
GEN$SHR_CEFCNT            000000CC RG     03
GEN$SHR_CEFMAX            000000EE RG     03
GEN$SHR_GBLCNT            000000AC RG     03
GEN$SHR_GBLMAX            000000DC RG     03
GEN$SHR_INIT              00000140 RG     03
GEN$SHR_MBXCNT            000000BC RG     03
GEN$SHR_MBXMAX            000000E5 RG     03
GEN$SHR_MEMNAME           00000091 RG     03
GEN$SHR_POOLC             000000F7 RG     03
GEN$SHR_POOLS             00000107 RG     03
GEN$SHR_PRQCNT            00000127 RG     03
GEN$SHR_RESET             0000004A RG     03
GEN$SHR_START             00000137 RG     03
GEN$SHR_UNIT              000000A3 RG     03
GSD$B_CREATPORT         = 00000052
GSD$B_DELETPORT         = 00000053
GSD$B_LOCK              = 00000050
GSD$B_PROCCNT           = 00000051
GSD$B_TYPE              = 0000000A
GSD$K_SHMGSDLNG         = 00000074
GSD$L_GSDBL             = 00000004
GSD$L_GSDFL             = 00000000
GSD$L_PTECNT1           = 00000074
GSD$V_VALID             = 00000000
GSD$W_GSTX              = 00000016
GSD$W_SIZE              = 00000008
INI$A[ONONPAGED           0000020F RG     03
INI$MPMADP                ********    X   03
INIT                      00000217 R      03
INITLOCK_TIMOUT         = 08F0D180
INITPOLL_TIMOUT         = 02FAF080
INIT_BITMAP               000005DD R      03
INIT_CEF                  0000068F R      03
INIT_DATAPAGE             000003E6 R      03
INIT_MAILBOXES            00000676 R      03
INIT_POOL                 00000635 R      03
INIT_PRQ                  00000658 R      03
INIT_STRUCTURES           000005A7 R      03
IOC$ALLOSPT               ********    X   03
IOC$GL_ADPLIST            ********    X   03
IOGEN$ALLOBLOCK           ********    X   03
IOGEN$LOADER              ********    X   03
IOGEN$TEST_MEM            ********    X   03
IPL$_HWCLK              = 00000018
LIB$PUT_OUTPUT            ********    X   03
LIB$SIGNAL                ********    X   03
LOADMBDRIVER              0000083C R      03
LOCK_DATAPAGE             0000035C R      03
MA$INITIAL                ********    X   03
MA$REQUEST                ********    X   03
MAP_DATAPAGE              000002CC R      03
MAP_STRUCTURES            00000562 R      03
MBX$B_CREATPORT         = 00000009
MBX$B_FLAGS             = 00000008
MBX$K_LENGTH            = 00000030
```

```
MBX$W_REF                       =  0000000C
MBX$W_UNIT                      =  0000000A
MPM$C_PORTS                     =  00000004
MPM$L_CSR                       =  00000000
MPM$L_INV                       =  0000000C
MPM$L_MR                        =  0000001C
MPM$S_CSR_PORT                  =  00000002
MPM$S_INV_MEMSZ                 =  00000003
MPM$S_INV_STADR                 =  0000000B
MPM$S_MR_UNIT                   =  00000002
MPM$V_CSR_PORT                  =  00000000
MPM$V_INV_MEMSZ                 =  00000010
MPM$V_INV_STADR                 =  00000014
MPM$V_MR_UNIT                   =  0000000E
NDT$_MPM0                       =  00000040
NDT$_MPM3                       =  00000043
NO_INIT                            000003E3 R      03
POLL                               0000039A R      03
PR$_IPL                         =  00000012
PR$_TBIS                        =  0000003A
PRQ$C_MAILBOX                   =  00000001
PRQ$C_MINLENGTH                 =  00000040
PTE$C_ERKW                      =  30000000
PTE$M_VALID                     =  80000000
PTE$S_PFN                       =  00000015
PTE$V_PFN                       =  00000000
RPB$L_BOOTR5                    =  00000030
RPB$L_MEMDSC                    =  000000BC
RPB$M_MPM                       =  00000800
RPB$M_USEMPM                    =  00001000
RSN$_MAX                        =  0000000F
SHARE                              00000185 R      03
SHB$B_FLAGS                     =  0000000B
SHB$B_NEXUS                     =  00000014
SHB$B_PORT                      =  00000015
SHB$B_TYPE                      =  0000000A
SHB$K_LENGTH                    =  00000020
SHB$L_ADP                       =  0000001C
SHB$L_BASGSPFN                  =  00000010
SHB$L_DATAPAGE                  =  00000004
SHB$L_LINK                      =  00000000
SHB$L_POOLEND                   =  00000018
SHB$L_REFCNT                    =  0000000C
SHB$M_CONNECT                   =  00000001
SHB$V_CONNECT                   =  00000000
SHB$W_SIZE                      =  00000008
SHD$B_FLAGS                     =  0000009F
SHD$B_INITLCK                   =  0000009D
SHD$B_PORTS                     =  0000009C
SHD$K_LENGTH                    =  00000180
SHD$L_CEFPTR                    =  00000008
SHD$L_CRC                       =  00000038
SHD$L_GSBITMAP                  =  0000000C
SHD$L_GSDPTR                    =  00000004
SHD$L_GSPAGCNT                  =  00000010
SHD$L_GSPFN                     =  00000014
SHD$L_MBXPTR                    =  00000000
SHD$M_INITLCK                   =  00000001
SHD$Q_INITTIME                  =  00000030
SHD$Q_POOL                      =  000000F8
SHD$Q_PRQ                       =  000000F0
SHD$Q_PRQWRK                    =  00000100
SHD$T_NAME                      =  00000020
SHD$V_INITLCK                   =  00000000
SHD$V_MBXLCK                    =  00000003
SHD$W_CEFMAX                    =  0000001C
SHD$W_CEFQUOTA                  =  0000007C
SHD$W_GSDMAX                    =  00000018
SHD$W_GSDQUOTA                  =  0000003C
SHD$W_MBXMAX                    =  0000001A
SHD$W_MBXQUOTA                  =  0000005C
SHD$W_POLL                      =  000000A6
SHD$W_PRQWAIT                   =  000000A4
SHD$W_RESAVAIL                  =  000000C8
SHD$W_RESSUM                    =  000000E8
SHD$W_RESWAIT                   =  000000A8
SHOW_STRUCT                        0000087F R      03
SHR_B_OPTIONS                      00000026 R      02
SHR_L_ADP                          0000003F R      02
SHR_L_CEFSIZE                      00000033 R      02
SHR_L_DATAPAGE                     00000037 R      02
SHR_L_GSDSIZE                      0000002F R      02
SHR_L_MEMPFN                       0000002B R      02
SHR_L_MEMSIZE                      00000027 R      02
SHR_L_POOLBCNT                     00000016 R      02
SHR_L_POOLBSIZ                     0000001A R      02
SHR_L_PRQCNT                       0000001E R      02
SHR_L_SHDPTE                       0000003B R      02
SHR_L_START                        00000022 R      02
SHR_OPT_M_INIT                  =  00000001
SHR_OPT_V_INIT                  =  00000000
SHR_Q_MEMNAME                      00000000 R      02
SHR_T_MBDEVNAME                    00000043 R      02
SHR_T_MBDRVNAME                    00000000 R      03
SHR_VALUES                         00000000 R      02
SHR_W_CEFCNT                       0000000E R      02
SHR_W_CEFQUO                       00000014 R      02
SHR_W_GBLCNT                       0000000A R      02
SHR_W_GBLQUO                       00000010 R      02
SHR_W_MBXCNT                       0000000C R      02
SHR_W_MBXQUO                       00000012 R      02
SHR_W_UNIT                         00000008 R      02
SIZ...                          =  00000001
SS$_NORMAL                      =  00000001
STS$K_ERROR                     =  00000002
STS$K_SUCCESS                   =  00000001
STS$S_SEVERITY                  =  00000003
STS$V_SEVERITY                  =  00000000
SYS$CMEXEC                         ********  GX    03
SYS$CMKRNL                         ********  GX    03
SYS$FAO                            ********  X     03
SYSG$_BADCHKSUM                 =  007C8052
SYSG$_BADPARAM                  =  007C804A
SYSG$_INCMEMNAM                 =  007C905A
```

```
SYSG$_NOSUCHMEM                 = 007C8042
SYSG$_SHMDBLUSE                 = 007C810A
SYSG$_SPTFULL                   = 007C8022
TPA$L_NUMBER                    = 0000001C
TPA$L_TOKENCNT                  = 00000010
UNLOCK_DATAPAGE                   0000038E R      03
UNLOCK_EXIT                       0000027A R      03
VA$M_SYSTEM                     = 80000000
VA$S_VPN                        = 00000015
VA$V_VPN                        = 00000009
```

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
NONPAGED_DATA                   000000CB  (  203.)  02 (  2.)  NOPIC   USR   CON   REL   LCL NOSHR NOEXE   RD    WRT NOVEC QUAD
NONPAGED_CODE                   000009B4  ( 2484.)  03 (  3.)  NOPIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC LONG
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.07	00:00:00.36
Command processing	134	00:00:00.68	00:00:04.53
Pass 1	456	00:00:18.00	00:00:33.60
Symbol table sort	0	00:00:02.18	00:00:04.24
Pass 2	259	00:00:04.50	00:00:09.51
Symbol table output	28	00:00:00.23	00:00:00.62
Psect synopsis output	1	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	915	00:00:25.70	00:00:52.90

```
The working set limit was 1800 pages.
102872 bytes (201 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1465 non-local and 71 local symbols.
1355 source lines were read in Pass 1, producing 24 object records in Pass 2.
44 pages of virtual memory were used to define 41 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
_255DUA28:[SYS.OBJ]LIB.MLB;1	24
_255DUA28:[SYSLIB]STARLET.MLB;2	13
TOTALS (all libraries)	37

```
1619 GETS were required to define 37 macros.
```

```
There were no errors, warnings or information messages.

MACRO/LIS=LIS$:SHARE/OBJ=OBJ$:SHARE MSRC$:SHARE/UPDATE=(ENH$:SHARE)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB
```

READLBN
LIS

RTFILREAD
LIS

SHARE
LIS

QUSS
LIS

RWBTDRIVR
LIS

MBBTDRIVR
LIS

SCSLOADER
LIS

RMSCONIO
LIS

READDRIV
LIS

PABTDRIVR
LIS

PUBTDRIVR
LIS

PUTERROR
LIS

READPRMPT
LIS

SYSBOOCMD
LIS

SHOWADAP
LIS

STANDCONF
LIS

SHODEV
LIS

STAPUTERR
LIS

STALOCK
LIS

STASGNMSG
LIS

STACONFIG
LIS

STASYSGEN
LIS

SYSBOOT
LIS

STARDBRIV
LIS