F 5

```
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
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         COO        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        SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO           TTT        SSSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO           TTT        SSSSSSSSSSSS
```

FILEID**CONFIG

```
  CCCCCCCC    OOOOOO    NN      NN  FFFFFFFFFF    IIIIII      GGGGGGGG
  CCCCCCCC    OOOOOO    NN      NN  FFFFFFFFFF    IIIIII      GGGGGGGG
CC          OO      OO  NN      NN  FF              II      GG
CC          OO      OO  NN      NN  FF              II      GG
CC          OO      OO  NNNN    NN  FF              II      GG
CC          OO      OO  NNNN    NN  FF              II      GG
CC          OO      OO  NN  NN  NN  FFFFFFFF        II      GG
CC          OO      OO  NN  NN  NN  FFFFFFFF        II      GG
CC          OO      OO  NN    NNNN  FF              II      GG  GGGGGG
CC          OO      OO  NN    NNNN  FF              II      GG  GGGGGG
CC          OO      OO  NN      NN  FF              II      GG      GG      ....
CC          OO      OO  NN      NN  FF              II      GG      GG      ....
  CCCCCCCC    OOOOOO    NN      NN  FF            IIIIII      GGGGGG        ....
  CCCCCCCC    OOOOOO    NN      NN  FF            IIIIII      GGGGGG        ....

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
```

```
0000            .TITLE  CONFIG - CSR AND VECTOR UTITLITY
0000            .IDENT  'V04-000'
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ;
0000    ; AUTHOR:  Jake VanNoy          Creation Date:  18-JAN-1981
0000    ;
0000    ; FACILITY:     BOOTS, SYSGEN
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-013 WHM0006         Bill Matthews           27-Jun-1984
0000    ;               Fix display of SHO/CONF and SHO/CONF/COMM for MicroVAX I.
0000    ;
0000    ;       V03-012 WHM0005         Bill Matthews           26-Mar-1984
0000    ;               Fixed linker truncation errors.
0000    ;
0000    ;       V03-011 WHM0004         Bill Matthews           16-Feb-1984
0000    ;               Added equivalence name IEU11 for IEQ11.
0000    ;               Added support for the new IDB field IDB$B_COMBO_VECTOR_OFFSET.
0000    ;
0000    ;       V03-010 WHM0003         Bill Matthews           02-Feb-1984
0000    ;               Added equivalence name DHU11 for DHV11.
0000    ;
0000    ;       V03-009 WHM0002         Bill Matthews           01-Feb-1984
0000    ;               Changed FAO parameter of CONNECT_OTHER from UL to UW.
0000    ;               Added support for a valid adapter 0.
0000    ;
0000    ;       V03-008 WHM0001         Bill Matthews           15-Dec-1983
0000    ;               Added support for outputting CONNECT qualifiers /VECTOR_OFFSET
0000    ;               and /CSR_OFFSET.
0000    ;
0000    ;       V03-007 MSH0005         Maryann Hinden          27-Jun-1983
```

```
0000    ;               Fix truncation error for call to CHECK_CSR.
0000    ;
0000    ;       V03-006 MSH0004         Maryann Hinden          24-Jun-1983
0000    ;               Change $BOODEF to $BOOCMDDEF.
0000    ;
0000    ;       V03-005 MSH0003         Maryann Hinden          23-Jun-1983
0000    ;               Use $BOODEF.
0000    ;
0000    ;       V03-004 MSH0002         Maryann Hinden          28-Dec-1982
0000    ;               Calculate floating vector for nth device (n>1)
0000    ;               correctly; add UNA and TU81 to REARNG table.
0000    ;
0000    ;       V03-003 MSH0001         Maryann Hinden          04-Oct-1982
0000    ;               Check for DDB$L_UCB = 0.
0000    ;
0000    ;       V03-002 KDM0002         Kathleen D. Morse       28-Jun-1982
0000    ;               Added $PRDEF.
0000    ;
0000    ; ABSTRACT:
0000    ;
0000    ;       CONFIG is a utility routine used to calculate the CSR and  vector
0000    ;       addresses  that  AUTOCONFIGURE would assign to a configuration on
0000    ;       the UNIBUS.  Input consists of a list of devices that make up any
0000    ;       possible  configuration  of devices on the UNIBUS.
0000    ;
0000    ;    Form of input: A file of <device type, # of, previous #> pairs.  The
0000    ;       ordering of the devices in this file is unimportant, the utility
0000    ;       will calculate ranking. The previous # is the count of this device
0000    ;       type that were configured on previous UNIBUS's.
0000    ;
0000    ;    Output: A list of CSR and  vector  addresses  that  AUTOCONFIGURE
0000    ;          would use for such a configuration.
0000    ;
0000    ;    Sample of input:
0000    ;
0000    ;          CR11
0000    ;          ! comments are allowed after "!"
0000    ;          LP11,2,2
0000    ;          DC11,6
0000    ;          DT11,,1
0000    ;          RL211,3
0000    ;
0000    ;          where  the absence of a number of controllers is taken to mean
0000    ;          that there is one such device.  Unrecognized device types will
0000    ;          be flagged as errors.  Use of equivalent names will be noted -
0000    ;          e.g.  the use of RL211 above will result in the line:
0000    ;
0000    ;             Equivalent Name - Device RL211 will be output as RL11.
0000    ;
0000    ;--
0000
```

```
                0000
                0000    ;
                0000    ; MACRO LIBRARY CALLS
                0000    ;
                0000
                0000            $ACFDEF                         ;DEFINE ACF OFFSETS
                0000            $ADPDEF
                0000            $BOOCMDDEF
                0000            $CRBDEF
                0000            $DCDEF
                0000            $DDBDEF
                0000            $DSCDEF
                0000            $IDBDEF
                0000            $PRDEF
                0000            $SSDEF
                0000            $SYSGMSGDEF
                0000            $TPADEF
                0000            $UCBDEF
                0000            $VECDEF
                0000
                0000    ;
                0000    ; CONSTANTS:
                0000    ;
                0000
      00001000  0000    UBA_IOBASE = 8*512
                0000
      00000020  0000    SPACE = ^X20
      00000080  0000    BUFFER_SIZE = 128
                0000
      00000000  0000    UBA_V_SUPPORT  = 0
      00000001  0000    UBA_M_SUPPORT  = 1
                0000
      00000001  0000    UBA_V_FLOATCSR = 1
      00000002  0000    UBA_M_FLOATCSR = 2
                0000
      00000002  0000    UBA_V_FLOATVEC = 2
      00000004  0000    UBA_M_FLOATVEC = 4
                0000
```

```
0000  .SBTTL EQUIVALENT NAMES MACRO
0000
0000  ;
0030  ; MACROS:
0000  ;
0000
0000          .MACRO  SIGNAL  message
0000
0000          .IF NB MESSAGE
0000          MOVL    Message,R0
0000          .ENDC
0000
0000          BSBW    SIGNAL_R0
0000
0000          .Endm   SIGNAL
0000
0000  ;
0000  ;
0000  ; macro to generate equivalences data structure
0000  ;
0000  ; This macro creates a tree-like data structure where
0000  ; each pair of nodes consists of two names. The first
0000  ; name is a name which appears in the autoconfigure table
0000  ; and the second name is another possible name for the
0000  ; same device.
0000  ; The two macros FIND_EQV are then used
0000  ; to give one the other node, given the first.
0000  ;
0000
0000          .MACRO  EQUIV   NAME1,NAME2
0000
0000                  .PSECT  ACF_NAMES
0000          $NAME1$=.
0000                  .ASCID  /NAME1/         ;name in ubatable
0000
0000                  .PSECT  EQV_NAMES
0000          $NAME2$=.
0000                  .ASCID  /NAME2/         ;equivalent name
0000
0000                  .PSECT  EQV_DESC
0000          $EQV_DESC$=.
0000                  .LONG   $NAME1$
0000                  .LONG   $NAME2$
0000
0000                  .PSECT  EQV_DATA
0000                  .LONG   $EQV_DESC$
0000
0000          .ENDM   EQUIV
0000
```

```
0000    .SBTTL REARNG; REARRANGE DEVICES ARRAY MACRO
0000
0000    ;
0000    ; macro to rearrange numbers in DEVICES array
0000    ; intended to handle exceptions like RL11
0000    ; where one device is fx,fx and another is fx,fl, etc.
0000    ;
0000    ;
0000    ; INPUT
0000    ;
0000    ;       R11   - address of DEVICES
0000    ;       FIRST - first name in ubatable (upper listing)
0000    ;       OCC1  - which occurance of FIRST to find
0000    ;       SECOND- second name in ubatable
0000    ;       OCC2  - which occurance of SECOND to find
0000    ;
0000    ; OUTPUT
0000    ;
0000    ;       The DEVICES array becomes:
0000    ;       DEVICES[second] := DEVICES[first] - 1
0000    ;       DEVICES[first] := 1
0000    ;
0000    ;-
0000
0000    .MACRO  REARNG  FIRST,OCC1,SECOND,OCC2 ,?L1
0000
0000
0000            PUSHAL  FIRST                           ; First device
0000            PUSHL   #OCC1                           ; Occurance
0000            PUSHAL  SECOND                          ; Second device
0000            PUSHL   #OCC2                           ; Occurance
0000            CALLS   #4,W^REARNG_DEV                 ; Rearrange
0000            BLBS    R0,L1                           ; Better not be error
0000            BRW     EXIT                            ; Will fail to run if so
0000    L1:
0000
0000            .ENDM   REARNG
0000
0000    ;+
0000    ;
0000    ; Macro to increment controller name by calling routine in AUTOCONFG
0000    ;
0000    ;-
0000
0000    .MACRO  INC_CHAR
0000
0000            PUSHL   R1                              ; Save R1
0000            MOVL    W^L_DEVNAME,R1                  ; Address of device name
0000            JSB     ACF$INC_CHAR                    ; Increment character, routine does not
0000                                                    ;       return status
0000            POPL    R1                              ; Restore R1
0000
0000    .ENDM   INC_CHAR
0000
0000
```

```
                                     0000   255 .SBTTL  SYMBOLS AND DATA AREA
                                     0000   256
                                 00000000   257 .PSECT  PAGED_DATA      rd,wrt,noexe,quad
                                     0000   258 ;
                                     0000   259 ; LOCAL VARIABLES:
                                     0000   260 ;
65 63 69 76 65 44 00000008'010E0000' 0000   261 DEV_LINE:   .ASCID @Device: !AC !_Name: !4<!AC!> !_ CSR: !6OL!AC !_ Vector:@-
65 6D 61 4E 5F 21 20 43 41 21 20 3A  000E
21 20 3E 21 43 41 21 3C 34 21 20 3A  001A
21 4C 4F 36 21 20 3A 52 53 43 20 5F  0026
72 6F 74 63 65 56 20 5F 21 20 43 41  0032
                                  3A 003E
53 5F 21 20 43 41 21 4C 4F 33 21 20  003F   262 @ !3OL!AC !_Support: !AC @
20 43 41 21 20 3A 74 72 6F 70 70 75  004B
                            56 53 52 0057   263 RSV:            .ASCII  /RSV/
                     31 31 4C 52 00' 005A   264 RL11:           .ASCIC  /RL11/
                                  04 005A
                  31 31 32 4C 52 00' 005F   265 RL211:          .ASCIC  /RL211/
                                  05 005F
                     31 31 53 54 00' 0065   266 TS11:           .ASCIC  /TS11/
                                  04 0065
                  31 31 32 58 52 00' 006A   267 RX211:          .ASCIC  /RX211/
                                  05 006A
                        41 44 55 00' 0070   268 UDA:            .ASCIC  /UDA/
                                  03 0070
                  31 31 41 50 4C 00' 0074   269 LPA11:          .ASCIC  /LPA11/
                                  05 0074
                  42 31 31 52 44 00' 007A   270 DR11B:          .ASCIC  /DR11B/
                                  05 007A
                        41 4E 55 00' 0080   271 UNA:            .ASCIC  /UNA/
                                  03 0080
                     31 38 55 54 00' 0084   272 TU81:           .ASCIC  /TU81/
                                  04 0084
                            000000ED 0089   273 DEVICES:        .BLKB   100             ;array of device counts as read in
                                     00ED   274
                            00000000 00ED   275 L_DEVNAME:      .LONG   0               ;addr. of device name
                            00000000 00F1   276 L_DRVNAME:      .LONG   0               ;addr. of driver name
                            00000000 00F5   277 L_ROUTINE:      .LONG   0               ;addr. of routine name
                            00000000 00F9   278 ACF_NAME:       .LONG   0               ;used in routine lookup
                            00000000 00FD   279 OFFSET:         .LONG   0
                            00000000 0101   280 NUM:            .LONG   0               ;number of devices
                                     0105   281
                              20 00' 0105   282 FX:             .ASCIC  / /
                                  01 0105
                              2A 00' 0107   283 FL:             .ASCIC  /*/
                                  01 0107
                        73 65 79 00' 0109   284 YES:            .ASCIC  /yes/
                                  03 0109
                           6F 6E 00' 010D   285 NO:             .ASCIC  /no/
                                  02 010D
                            00000000 0110   286 SUP:            .LONG   0
                            00000000 0114   287 W_CSRBASE:      .LONG   0               ;addr. of current csr
                            00000000 0118   288 W_VECBASE:      .LONG   0               ;addr. of current vector
                                  00 011C   289 B_CNUMVEC:      .BYTE   0               ;number of vectors
                                0000 011D   290 W_VECMOD:       .WORD   0               ;floating vector modulus
                                     011F   291 OUTPUT_DESC:
                  00000127'010E0000' 011F   292 UPCASE_SRC:     .ASCID  //              ;only need ascid block
```

```
                 0000012F'010E0000' 0127   293 UPCASE_DST:     .ASCID  //              ;has to BUFFER_SIZE long
                          000001AF  012F   294                 .BLKB   BUFFER_SIZE     ;here's the block
                                    01AF   295
45 43 49 56 45 44 000001B7'010E0000' 01AF   296 CONF_PR:        .ASCID  /DEVICE> /
                             20 3E  01BD
                                    01BF   297
                          FFFFFFFF  01BF   298 BOO$GL_TR::     .LONG   -1              ; For /ADAPTER =
                                    01C3   299
                 000001CB'010E0000' 01C3   300 null:           .ascid  //
                                    01CB   301 fao_d_outbuf:
                          00000080  01CB   302                         .long   128
                          000001D3' 01CF   303                         .long   buf
                          00000253  01D3   304 buf:                    .blkb   128
                                    0253   305 fao_d_ctrstr:
65 72 64 64 41 20 0000025B'010E0000' 0253   306           .ASCID  @ Address !6OL (!XL) responds with value !XW (hex) @
4C 58 21 28 20 4C 4F 36 21 20 73 73  0261
77 20 73 64 6E 6F 70 73 65 72 20 29  026D
58 21 20 65 75 6C 61 76 20 68 74 69  0279
            20 29 78 65 68 28 20 57  0285
                                    028D   307 fao_q_oneuba:
2A 20 2F 21 2F 21 00000295'010E0000' 028D   308           .ascid  @!/!/ ** UNIBUS map for nexus #!UL on !%D ** @
70 61 6D 20 53 55 42 49 4E 55 20 2A  029B
23 20 73 75 78 65 6E 20 72 6F 66 20  02A7
2A 20 44 25 21 20 6E 6F 20 4C 55 21  02B3
                             20 2A  02BF
                              0000  02C1   309 fao_w_outlen:           .word   0
                                    02C3   310
```

```
          02C3   .SBTTL  TPARSE TABLE FOR CONFIG INPUT LINE
          02C3
          02C3   $INIT_STATE     CNF$STATE,CNF$KEYTBL
          02C3
          02C3   $STATE
          02C3   $TRAN    !DEVICE,TPA$_EX!T               ; Parse device line
          02C3   $TRAN    '!',TPA$_EXIT                   ; Allow comment
          02C3   $TRAN    TPA$_BLANK,TPA$_EXIT            ; Allow non-null line with only blanks
          02C3
          02C3   $STATE  DEVICE
          02C3   $TRAN    TPA$_STRING,NUMBER,CNF$FIND_DEVICE
          02C3
          02C3   $STATE  NUMBER
          02C3   $TRAN    <','>                           ; Accept ","
          02C3   $TRAN    TPA$_LAMBDA                     ; But make it optional
          02C3   $STATE
          02C3   $TRAN    TPA$_DECIMAL,NUMBER2,CNF$SET_VALUE ; Allow device,m,n
          02C3   $TRAN    TPA$_LAMBDA,NUMBER2,CNF$SET_VALUE ; Allow device,,n
          02C3
          02C3
          02C3   $STATE  NUMBER2                          ; Previous UNIBUS count
          02C3   $TRAN    <','>
          02C3   $TRAN    TPA$_LAMBDA
          02C3   $STATE
          02C3   $TRAN    TPA$_DECIMAL,TPA$_EXIT,CNF$PREV_UNIBUS ; Second # is prev unibus dev count
          02C3   $TRAN    TPA$_LAMBDA,TPA$_EXIT
          02C3
          02C3   $END_STATE
          02C3
000002E7  02C3   PARAM_BLK:      .BLKB   TPA$K_LENGTH0
```

```
                  02E7  .SBTTL EQV_TABLE DATA
                  02E7
              00000000  .PSECT  EQV_DATA
                  0000
                  0000   ;
                  0000   ; Data for equivalences table. First device name is as it appears in the
                  0000   ; autoconfigure table. Second device is a possible second name (for whatever
                  0000   ; reason). The program uses this table to allow either device name as input.
                  0000   ;
                  0000
                  0000   AB_EQV_TABLE::
                  0000
                  0000   ;               acf      eqv     why
                  0000   ;               ----     -----   ---------------------------
                  0000          EQUIV   RK611   ,RK711   ;multiple names
                  0004
                  0004          EQUIV   DZ11    ,DZ32    ;multiple names
                  0008
                  0008          EQUIV   DR11C   ,DR11A   ;multiple names
                  000C
                  000C          EQUIV   DL11C   ,DL11D   ;multiple names
                  0010          EQUIV   DL11C   ,DL11E   ;multiple names
                  0014
                  0014          EQUIV   RL11    ,RL211   ;multiple names
                  0018
                  0018          EQUIV   RX211   ,RX02    ;multiple names
                  001C
                  001C          EQUIV   DR11W   ,XA11    ;multiple names
                  0020          EQUIV   DR11W   ,DR11    ;because of chapter 14 How To Write...
                  0024          EQUIV   DHV11   ,DHU11   ;multiple names
                  0028          EQUIV   IEQ11   ,IEU11   ;multiple names
                  002C
       00000000   002C          .LONG   0                ;end of list
```

```
                                 0030  .SBTTL  TPARSE ACTION ROUTINES
                                 0030
                             00000000  .PSECT  PAGED_CODE      rd,nowrt,exe,long
                                 0000
                                 0000  ;+
                                 0000  ;
                                 0000  ; These are the TPARSE action routines called by the parsing table
                                 0000  ; within the CONFIGURE module and from SYSBOOCMD. The routines called
                                 0000  ; from SYSBOOCMD are prefixed by BOO$, the routines called by CONFIGURE's
                                 0000  ; TPARSE table are prefixed by CNF$, other routines in this module that
                                 0000  ; are called as subroutines have no prefix.
                                 0000  ;
                                 0000  ;-
                                 0000
                           0000  0000  .ENTRY  BOO$NO_RESET, ^M<>
                                 0002
00 00000000'EF    01        E2   0002          BBSS    #BOOCMD$V_NORESET,BOO$GL_CMDOPT,10$ ; Set /NORESET bit1
                            04   000A  10$:    RET
                                 000B
                           0000  000B  .ENTRY  BOO$SET_TR, ^M<>
                                 000D
000001BF'EF    1C AC        D0   000D          MOVL    TPA$L_NUMBER(AP),BOO$GL_TR      ; Set /ADAPTER = number
                            04   0015          RET
                                 0016
```

```
                                   0016   402 ;+
                                   0016   403 ;
                                   0016   404 ; This is a TPARSE action routine that offsets the device controller
                                   0016   405 ; character to allow correct device names on mutiple UNIBUS configurations
                                   0016   406 ;
                                   0016   407 ;-
                                   0016   408
                             000C  0016   409 .ENTRY  CNF$PREV_UNIBUS, ^M<R2,R3>
                                   0018   410
          52   00FD'CF       D0    0018   411         MOVL    W^OFFSET,R2              ; Offset of this device into UBATABLE
    52   00000000'EF42       DE    001D   412         MOVAL   L^ACF$AB_UBATABLE[R2],R2 ; Address in table
                    53       D4    0025   413         CLRL    R3                       ; Index
                                   0027   414
          51    00 B2        D0    0027   415 10$:    MOVL    @(R2),R1                 ; Address of ascic device name string
         00000000'EF         16    002B   416         JSB     ACF$INC_CHAR             ; Increment character
        F1 53   1C AC        F2    0031   417         AOBLSS  TPA$L_NUMBER(AP),R3,10$ ; Add one and branch while LSS
             50    01        D0    0036   418         MOVL    #1,R0                    ; Set success
                             04    0039   419         RET                              ; Return
                                   003A   420
                                   003A   421 ;
                                   003A   422 ; routine to set value in device array
                                   003A   423 ;
                                   003A   424
                             00FC  003A   425 .ENTRY  CNF$SET_VALUE, ^M<R2,R3,R4,R5,R6,R7>
                                   003C   426
           52   1C AC        F6    003C   427         CVTLB   TPA$L_NUMBER(AP),R2      ; Convert to byte
                    31       1D    0040   428         BVS     20$                      ; Overflow error
                    2F       15    0042   429         BLEQ    20$                      ; zero or negative
                                   0044   430
          53   0089'CF       9E    0044   431         MOVAB   W^DEVICES,R3             ; Base of array
          54   00FD'CF       D0    0049   432         MOVL    W^OFFSET,R4              ; Offset into array
                  6344       95    004E   433         TSTB    (R3)[R4]                 ; Make sure its zero
                    1A       13    0051   434         BEQL    10$                      ; Branch if OK
                                   0053   435
                                   0053   436 ;       device has been input by user twice -
                                   0053   437 ;       replace original value and notify user.
                                   0053   438
                1C AC        DD    0053   439         PUSHL   TPA$L_NUMBER(AP)         ; Second number for this device
           7E   6344         9A    0056   440         MOVZBL  (R3)[R4],-(SP)           ; First number that was input
              00ED'CF        DD    005A   441         PUSHL   W^L_DEVNAME              ; Name of device
                   03        DD    005E   442         PUSHL   #3                       ; Number of FAO parameters
            007CA00B 8F      DD    0060   443         PUSHL   #SYSG$_TWICE             ; Error message
       00000000'GF   05      FB    0066   444         CALLS   #5,G^LIB$SIGNAL          ; Signal error
                                   006D   445
           6344    52        90    006D   446 10$:    MOVB    R2,(R3)[R4]              ; Insert number of devices into array
                   0A        11    0071   447         BRB     30$                      ; Return success
                                   0073   448
                                   0073   449 20$:
                                   0073   450         SIGNAL  #SYSG$_OUT_RANGE         ; message - warning
                                   007D   451
             50    01        D0    007D   452 30$:    MOVL    #1,R0                    ; Set success
                             04    0080   453         RET                              ; Return
```

```
                                 00FC  0081   455 .ENTRY  CNF$FIND_DEVICE, ^M<R2,R3,R4,R5,R6,R7>
                                       0083   456
                                       0083   457 ; Check first for equivalence name
                                       0083   458
         54   00000000'EF    9E  0083   459         MOVAB   AB_EQV_TABLE,R4         ; Address of equivalences table
                                       008A   460
                     55   84    D0  008A   461 10$:    MOVL    (R4)+,R5                ; Get next equivalence
                          3C    13  008D   462         BEQL    20$                     ; End of list - no equivalence found
                                       008F   463
              00F9'CF   85    D0  008F   464         MOVL    (R5)+,W^ACF_NAME        ; Possible real device name
                     56   85    D0  0094   465         MOVL    (R5)+,R6                ; Equivalence name to match against
  66    20    14 BC   10 AC    2D  0097   466         CMPC5   TPA$L_TOKENCNT(AP),@TPA$L_TOKENPTR(AP), -
                       04 B6        009E
                                       00A0   467                 #^A/ /,(R6),@4(R6)      ; Check for match
                          E8    12  00A0   468         BNEQ    10$                     ; Branch if not
                                       00A2   469
              56   00F9'CF    D0  00A2   470         MOVL    W^ACF_NAME,R6           ; Address of real device name
                                       00A7   471
                                       00A7   472 ; signal change to ACF_NAME
                                       00A7   473
                       04 A6    DD  00A7   474         PUSHL   4(R6)                   ; Address of ACF_NAME
                    7E    66    3C  00AA   475         MOVZWL  (R6),-(SP)              ; Length of ACF_NAME
                       14 AC    DD  00AD   476         PUSHL   TPA$L_TOKENPTR(AP)      ; Address of equivalence name
                       10 AC    DD  00B0   477         PUSHL   TPA$L_TOKENCNT(AP)      ; Length of equivalence name
                          04    DD  00B3   478         PUSHL   #4                      ; Number of FAO params
              007CA003 8F    DD  00B5   479         PUSHL   #SYSG$_EQV_NOTICE       ; Message name
        00000000'GF    06    FB  00BB   480         CALLS   #6,G^LIB$SIGNAL
                                       00C2   481
                10 AC    66    3C  00C2   482         MOVZWL  (R6),TPA$L_TOKENCNT(AP) ; Move in real count
          14 AC    04 A6    D0  00C6   483         MOVL    4(R6),TPA$L_TOKENPTR(AP) ; Move in real device name address
                                       00CB   484
                                       00CB   485 ;
                                       00CB   486 ; Find the string in UBATABLE
                                       00CB   487 ;
                                       00CB   488
                                       00CB   489
                       14 AC    DD  00CB   490 20$:    PUSHL   TPA$L_TOKENPTR(AP)      ; Push address of string to match
                       10 AC    DD  00CE   491         PUSHL   TPA$L_TOKENCNT(AP)      ; Push length of string
                          01    DD  00D1   492         PUSHL   #1                      ; Find first occurance
              00DF'CF    03    FB  00D3   493         CALLS   #3,W^LOOKUP             ;       in UBATABLE
                    03 50    E9  00D8   494         BLBC    R0,30$                  ; Branch if error
                                       00DB   495
                    50    01    D0  00DB   496         MOVL    #1,R0                   ; Return success
                                04  00DE   497 30$:    RET                             ;
                                       00DF   498
                                 00DF     .SBTTL  ROUTINE LOOKUP
                                 00DF
                                 00DF     ;+
                                 00DF     ;
                                 00DF     ; FUNCTIONAL DESCRIPTION
                                 00DF     ;
                                 00DF     ;       This routine locates a device name in UBATABLE
                                 00DF     ;
                                 00DF     ; CALLING SEQUENCE
                                 00DF     ;
                                 00DF     ;       PUSHL   <address of string to match against>
                                 00DF     ;       PUSHL   <length of string>
                                 00DF     ;       PUSHL   <which occurance of string to get>
                                 00DF     ;       CALLS   #3,LOOKUP
                                 00DF     ;
                                 00DF     ; INPUT PARAMETERS
                                 00DF     ;
                                 00DF     ;       as above
                                 00DF     ;
                                 00DF     ; OUTPUT PARAMETERS
                                 00DF     ;
                                 00DF     ;       R0      Completion code
                                 00DF     ;       OFFSET  Offset count into UBATABLE for device
                                 00DF     ;
                                 00DF     ;-
                                 00DF
                                 00DF     ; CONSTANTS
                                 00DF
                       00000004  00DF     OCCURANCE = 4
                       00000008  00DF     LENGTH    = 8
                       0000000C  00DF     ADDRESS   = 12
                                 00DF
                           007C  00DF     .ENTRY  LOOKUP, ^M<R2,R3,R4,R5,R6>
                                 00E1
                54    01     CE  00E1             MNEGL   #1,R4                     ; Initialize counter
          55   00000000'EF   9E  00E4             MOVAB   L^ACF$AB_UBATABLE,R5      ; Base address
                                 00EB
                56    85     D0  00EB     10$:    MOVL    (R5)+,R6                  ; Next device in table
                      27     13  00EE             BEQL    20$                       ; device not found - error
                      54     D6  00F0             INCL    R4                        ; Increment counter
             51    08 B6     9E  00F2             MOVAB   @8(R6),R1                 ; Address of device name string
                50    61     9A  00F6             MOVZBL  (R1),R0                   ; length of string
           00ED'CF    51     D0  00F9             MOVL    R1,W^L_DEVNAME            ; Save device name
  08 AC    20  01 A1  50     2D  00FE             CMPC5   R0,1(R1),#^A/ /, -
                   0C BC         0105
                                 0107                     LENGTH(AP),@ADDRESS(AP) ; Match ?
                      E2     12  0107             BNEQ    10$                       ; Branch if not
                                 0109
                   04 AC     D7  0109             DECL    OCCURANCE(AP)             ; Decrement occurances
                      DD     12  010C             BNEQ    10$                       ; Not this one, keep looking
                                 010E
                50    01     D0  010E             MOVL    #1,R0                     ; Return success
           00FD'CF    54     D0  0111             MOVL    R4,W^OFFSET               ; Save offset into table
                             04  0116             RET                               ; Return
                                 0117
                                 0117     20$:    ; signal no match
                                 0117
                   0C AC    DD  0117    556          PUSHL   ADDRESS(AP)              ; Address of unknown device string
                   08 AC    DD  011A    557          PUSHL   LENGTH(AP)               ; Length of string
                      02    DD  011D    558          PUSHL   #2                       ; Number of FAO arguments
           007C9008 8F      DD  011F    559          PUSHL   #SYSG$_DEVNOTKNWN        ; Device not known message
    00000000'GF       04    FB  0125    560          CALLS   #4,G^LIB$SIGNAL          ; SIGNAL error
    50     007C9008 8F      D0  012C    561          MOVL    #SYSG$_DEVNOTKNWN,R0     ; Set error so TPARSE will not continue
                            04  0133    562 30$:     RET                              ; Return
                                0134    563
```

```
                                   0134  .SBTTL  BOO$CONFIGURE - HYPOTHETICAL CONFIGURATION
                                   0134
                                   0134  ;++
                                   0134  ;
                                   0134  ; ABSTRACT:
                                   0134  ;
                                   0134  ;       BOO$CONFIGURE is the main TPARSE action routine called from SYSBOOCMD.
                                   0134  ;
                                   0134  ; INPUT:
                                   0134  ;
                                   0134  ;       OUTNAM_ADDR     address of ascii output file spec (Default = SYS$OUTPUT)
                                   0134  ;       OUTNAM_SIZE     size of ascii string
                                   0134  ;       INNAM_ADDR      address of ascii input file spec (Default = SYS$INPUT)
                                   0134  ;       INNAM_SIZE      size of ascii string
                                   0134  ;
                                   0134  ;--
                                   0134
                             0FFC  0134  .ENTRY  BOO$CONFIGURE, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                   0136
                                   0136  ; Open output file
                                   0136
                      FEC7'   30   0136          BSBW    BOO$OPEN_OUTPUT_2       ; Open /output= file (D=sys$output)
                    06 50     E8   0139          BLBS    R0,20$                  ; branch if no error
                                   013C
                                   013C  10$:    SIGNAL                          ; Signal
                       0043   31   013F          BRW     50$                     ; Exit on error
                                   0142
                                   0142  ; Open input file
                                   0142
                      FEBB'   30   0142  20$:    BSBW    BOO$OPEN_INPUT_2        ; Open /input= file (D=sys$input)
                    F4 50     E9   0145          BLBC    R0,10$                  ; Branch if error
                                   0148
       06 00000000'EF    01   E0   0148  40$:    BBS     #BOOCMD$V_NORESET,BOO$GL_CMDOPT,45$ ; Branch if /NORESET
             00000000'EF      16   0150          JSB     IOC$AUTORESET           ; Reset names
                                   0156
0064 8F    00   0089'CF  00   2C   0156  45$:    MOVC5   #0,W^DEVICES,#0,#100,W^DEVICES ; Zero device array
                0089'CF            015F
                                   0162
           56   011F'CF       DE   0162          MOVAL   W^UPCASE_SRC,R6         ; Set up calls to STR$UPCASE
           66   0080 8F       B0   0167          MOVW    #BUFFER_SIZE,(R6)       ;     made in READ-PARSE_INPUT
        04 A6   0000'CF       DE   016C          MOVAL   W^RIO$AB_INBUFFER,4(R6) ; Address in descriptor
           58   0127'CF       DE   0172          MOVAL   W^UPCASE_DST,R8         ; Destination descritor
           68   0080 8F       B0   0177          MOVW    #BUFFER_SIZE,(R8)       ; Length (Address already filled in)
                                   017C
                       000A   30   017C          BSBW    READ_PARSE_INPUT        ; Handle user input
                    03 50     E9   017F          BLBC    R0,50$                  ; Branch on error
                       00CD   30   0182          BSBW    ADDRESS_CALC            ; Process input
                                   0185
                 50   01      D0   0185  50$:    MOVL    #1,R0                   ; Set success for tparse
                              04   0188          RET                             ; Return to SYSBOOCMD
                                   0189
```

```
                          0189   616 READ_PARSE_INPUT:
                          0189   617
   20   0000'CF   00   2C 0189   618         MOVC5   #0,W^RIO$AB_INBUFFER,#^A/ /,-
  0000'CF   0080 8F       018F   619                 #BUFFER_SIZE,W^RIO$AB_INBUFFER ; Blank buffer
                          0195   620
                0E   E1   0195   621         BBC     #BOOCMD$V_INPUT,-
     1A 00000000'EF       0197   622                 BOO$GL_CMDOPT,5$        ; Branch if /INPUT not specified
                          019D   623         $GET    RAB=RIO_INRAB2
             20 50   E9   01AA   624         BLBC    R0,7$
        00000000'EF  B5   01AD   625         TSTW    RIO_INRAB2+RAB$W_RSZ    ; Test for zero bytes read in
                04   13   01B3   626         BEQL    READ_PARSE_INPUT        ; Read another record
                27   11   01B5   627         BRB     20$                     ; Branch
                          01B7   628 5$:
        000001AF'EF  9F   01B7   629         PUSHAB  CONF_PR                 ; Addess of prompt
                56   DD   01BD   630         PUSHL   R6                      ; Push address
  00000000'GF   02   FB   01BF   631         CALLS   #2,G^LIB$GET_INPUT      ; Get input
             04 50   E9   01C6   632         BLBC    R0,7$
                BE   13   01C9   633         BEQL    READ_PARSE_INPUT        ; Branch if zero
                11   11   01CB   634         BRB     20$
                          01CD   635
                          01CD   636 7$:
  00000000'8F   50   D1   01CD   637         CMPL    R0,#RMS$_EOF            ; End of file ?
                04   12   01D4   638         BNEQ    10$                     ; Branch if not
         50     01   D0   01D6   639         MOVL    #1,R0                   ; Set success
                     05   01D9   640         RSB                             ; Return
                          01DA   641
                          01DA   642 10$:
                          01DA   643         SIGNAL                          ; Signal
                     05   01DD   644         RSB                             ; Return
                          01DE   645
                          01DE   646 20$:
                          01DE   647
                          01DE   648 ; Must do upcasing since RMS only supports CVT in the ROP field for tty input
                          01DE   649
                56   DD   01DE   650         PUSHL   R6                      ; Source of UPCASE
                58   DD   01E0   651         PUSHL   R8                      ; Destination of UPCASE
  00000000'GF   02   FB   01E2   652         CALLS   #2,G^STR$UPCASE         ; Upcase input string
             04 50   E8   01E9   653         BLBS    R0,30$                  ; Branch if no error
                          01EC   654
                          01EC   655         SIGNAL                          ; Signal error
                     05   01EF   656         RSB                             ; Return
                          01F0   657
     57   02C3'CF    DE   01F0   658 30$:    MOVAL   W^PARAM_BLK,R7          ; Parameter block for tparse
             67     08   D0   01F5   659         MOVL    #TPA$K_COUNT0,TPA$L_COUNT(R7) ; # of longwords in param block
          04 A7     02   C8   01F8   660         BISL    #TPA$M_ABBREV,TPA$L_OPTIONS(R7)
                          01FC   661                                         ; Allow abbreviations, parse blanks
 08 A7   00000080 8F  D0   01FC   662         MOVL    #BUFFER_SIZE,TPA$L_STRINGCNT(R7) ; Size of BUFFER
      0C A7    04 A8  D0   0204   663         MOVL    4(R8),TPA$L_STRINGPTR(R7) ; Address of BUFFER
         1C A7     01  D0   0209   664         MOVL    #1,TPA$L_NUMBER(R7)     ; Default of one device
                          020D   665
          0000'CF    9F   020D   666         PUSHAB  W^CNF$KEYTBL            ; Set up for call to TPARSE
          0000'CF    9F   0211   667         PUSHAB  W^CNF$STATE             ;  by putting key table and state
                57   DD   0215   668         PUSHL   R7                      ;  and the param blk on stack
  00000000'GF   03   FB   0217   669         CALLS   #3,G^LIB$TPARSE         ; Call TPARSE
             03 50   E9   021E   670         BLBC    R0,40$                  ; Branch on error
              FF65   31   0221   671         BRW     READ_PARSE_INPUT        ; Continue to read input
                          0224   672
 007C8082 8F     50   D1  0224   673 40$:    CMPL    R0,#SYSG$_ABORT          ; Has a fatal error occurred ?
                 01   12  022B   674         BNEQ    50$                      ; Branch if not
                      05  022D   675         RSB                              ; Return on error
                          022E   676
 00000000'8F     50   D1  022E   677 50$:    CMPL    R0,#LIB$_SYNTAXERR       ; TPARSE error ?
                 03   13  0235   678         BEQL    60$                      ; Branch if not
               FF4F   31  0237   679         BRW     READ_PARSE_INPUT         ; Other errors already signaled
                          023A   680
                          023A   681 60$:
              14 A7   DD  023A   682         PUSHL   TPA$L_TOKENPTR(R7)       ; token that couldn't be parsed
              10 A7   DD  023D   683         PUSHL   TPA$L_TOKENCNT(R7)       ; length of token
                 02   DD  0240   684         PUSHL   #2                       ; Number of FAO params
        007C809A 8F   DD  0242   685         PUSHL   #SYSG$_SYNTAX            ; Error message
 00000000'GF     04   FB  0248   686         CALLS   #4,G^LIB$SIGNAL          ; Signal the error
               FF37   31  024F   687         BRW     READ_PARSE_INPUT
                          0252   688
                          0252  .SBTTL ROUTINE ADDRESS_CALC
                          0252
                          0252  ADDRESS_CALC:
                          0252  ;
                          0252  ; Make modifications to devices array and then simulate AUTOCONFIGURE
                          0252  ; algorithm by running through same decision process as it does.
                          0252  ; the difference is this code uses an array (devices) to tell what
                          0252  ; devices are there rather than the EXE$TEST_CSR routine to actually
                          0252  ; see if the device is physically present.
                          0252  ;
                          0252
                          0252  ;
                          0252  ; execeptions in devices array
                          0252  ;
                          0252  ; some modifications must be made to the devices array
                          0252  ; to allow proper positioning of devices
                          0252  ;
                          0252  ; all these special cases are handled here
                          0252  ;
                          0252
                          0252  ; the first exeception is the RSV device that, although it
                          0252  ; exists in the ubatable and is legal input as define above,
                          0252  ; it is a placeholder (a non-device) and cannot exist.
                          0252  ; therefore, the place in the devices array is checked to assure
                          0252  ; it is zero
                          0252
5B   0089'CF          DE  0252          MOVAL   W^DEVICES,R11
                          0257
     0057'CF          DF  0257          PUSHAL  W^RSV                   ; Address of string
          03          DD  025B          PUSHL   #3                      ; Length of string
          01          DD  025D          PUSHL   #1                      ; find first occurance
FE7B CF   03          FB  025F          CALLS   #3,LOOKUP               ;  in UBATABLE
57   00FD'CF          D0  0264          MOVL    W^OFFSET,R7             ; Offset of device into UBATABLE
                          0269
        6B47          95  0269          TSTB    (R11)[R7]               ; Must be zero
          0D          13  026C          BEQL    10$                     ; Since RSV is a non-device
                          026E                                          ; (.i.e just a placeholder)
                          026E
                          026E          SIGNAL  #SYSG$_RSV_ERR          ; Signal message
        6B47          94  0278          CLRB    (R11)[R7]               ; Zero out RSV
                          027B
                          027B  ;
                          027B  ; the following devices (RL11, TS11, LPA11, DR11B) all have
                          027B  ; a single fixed, fixed (or fixed, floating) address with
                          027B  ; the remainder as fixed, floating (or floating, floating).
                          027B  ;
                          027B  ; REARNG checks the first position and if it as greater than
                          027B  ; one shifts the balance to the second (where they should be)
                          027B  ;
                          027B
                          027B  10$:    REARNG  RL11,  1,       RL11,   2
                          0296          REARNG  TS11,  1,       TS11,   2
                          02B1          REARNG  LPA11, 1,       LPA11,  2
                          02CC          REARNG  UDA,   1,       UDA,    2
                          02E7          REARNG  RX211, 1,       RX211,  2
                          0302          REARNG  DR11B, 1,       DR11B,  2
                          031D          REARNG  DR11B, 2,       DR11B,  3       ; special case
```

```
0338   747        REARNG  UNA,   1,      UNA,   2
0353   748        REARNG  TU81,  1,      TU81,  2
```

```
                          036E
                          036E   ;++
                          036E   ;
                          036E   ; devices array is now in final form and ready to be interpretted
                          036E   ; as a list of devices on a UNIBUS. The following code keeps track
                          036E   ; of the CSR and vector addresses as it scans through the devices array.
                          036E   ; The devices array is in sorted order since it follows the ordering in
                          036E   ; the autoconfigure table.
                          036E   ;
                          036E   ;--
                          036E
 59   00000000'EF    9E   036E              MOVAB    L^ACF$AB_UBATABLE,R9     ; Initialize
0114'CF    E008 8F   B0   0375              MOVW     #^0160010,W^W_CSRBASE    ; Starting csr(minus high order bits)
0118'CF    00C0 8F   B0   037C              MOVW     #^0300   ,W^W_VECBASE    ; Starting vector
         57   01     CE   0383              MNEGL    #1,R7                    ; Counter (index into devices)
                          0386
         5A   89     D0   0386   LOOP:      MOVL     (R9)+,R10                ; Get first device addr.
              03     12   0389              BNEQ     10$                      ; Zero at end of list
            0190     31   038B              BRW      EXIT                     ; Exit
                          038E
00ED'CF    0000'CA   D0   038E   10$:       MOVL     UBT$L_DEVNAME(R10),W^L_DEVNAME  ; Address of device name
00F1'CF    0000'CA   D0   0395              MOVL     UBT$L_DRVNAME(R10),W^L_DRVNAME  ; Address of driver name
00F5'CF    0000'CA   D0   039C              MOVL     UBT$L_RTNNAME(R10),W^L_ROUTINE  ; Address of routine name
011C'CF    0000'CA   90   03A3              MOVB     UBT$B_NUMVEC(R10),W^B_CNUMVEC   ; Number of vectors
     53    0000'CA   9E   03AA              MOVAB    UBT$B_FLAGS(R10),R3             ; Byte flag
     5A    0000'CA   9E   03AF              MOVAB    UBT$W_REMAINDER(R10),R10        ; Remainder of UBADEV entry
                          03B4
              57     D6   03B4              INCL     R7                       ; Increment counter
   0101'CF  6B47     9A   03B6              MOVZBL   (R11)[R7],W^NUM          ; Get count of this device
                          03BC
                          03BC   ;
                          03BC   ; now determine if device is (fixed csr, fixed vector) or
                          03BC   ; (fixed csr,floating vector), or (floating csr, floating vector)
                          03BC   ; much of this code was extracted (and massaged) from AUTOCONFIGURE
                          03BC   ;
                          03BC   ; R1 used for vector addresses
                          03BC   ; R2 used for CSR addresses
                          03BC   ;
      1C 63   02     E1   03BC              BBC      #UBA_V_FLOATVEC,(R3),FX_FX ; Branch if not floating vector
                          03C0
                          03C0   ; device has a floating vector
                          03C0   ; round up to next valid vector address
                          03C0   ; boundary for next device in the table
                          03C0   ; store in R1
                          03C0
         51   6A     3C   03C0              MOVZWL   (R10),R1                 ; Get vector modulo mask
0000011D'EF   51     B0   03C3              MOVW     R1,W_VECMOD              ; Save for later
     51   0118'CF    A0   03CA              ADDW     W^W_VECBASE,R1           ; Round up
         51   8A     AA   03CF              BICW     (R10)+,R1                ; Truncate to actual vector offset
      03 63   01     E1   03D2              BBC      #UBA_V_FLOATCSR,(R3),20$ ; If clear - fixed CSR
            00BF     31   03D6              BRW      FL_FC
            0045     31   03D9   20$:       BRW      FX_FL
                          03DC
```

```
                           03DC
                           03DC  FX_FX:  ;fixed csr/fixed vector
                           03DC
       52    8A     3C     03DC  10$:    MOVZWL  (R10)+,R2             ; Get csr offset
             0C     12     03DF          BNEQ    30$                   ; Zero at end of fixed addresses
        0101'CF     D5     03E1          TSTL    W^NUM                 ; Better be zero
             03     13     03E5          BEQL    20$                   ; Branch if OK
           011F     31     03E7          BRW     TOO_MANY              ; Error - fatal
           FF99     31     03EA  20$:    BRW     LOOP
                           03ED
       51    8A     3C     03ED  30$:    MOVZWL  (R10)+,R1             ; Get vector address
        0101'CF     D5     03F0          TSTL    W^NUM                 ; Any of this type device left?
             03     12     03F4          BNEQ    40$
           FF8D     31     03F6          BRW     LOOP                  ; No device - no effect
    00000105'EF     DF     03F9  40$:    PUSHAL  FX
    00000105'EF     DF     03FF          PUSHAL  FX
   0549'CF   02     FB     0405          CALLS   #2,W^PUT_LINE         ; Output one line for this device
                           040A          INC_CHAR                      ; Increment controller
        0101'CF     D7     041A          DECL    W^NUM                 ; Decrement number
           FFBB     31     041E          BRW     10$                   ; Loop
                           0421
                           0421
                           0421  FX_FL:  ;fixed csr/floating vector
                           0421
       52    8A     3C     0421  10$:    MOVZWL  (R10)+,R2             ; Get fixed csr from ubatable
             03     12     0424          BNEQ    20$                   ; Zero indicates end of list
           004F     31     0426          BRW     40$                   ; Branch to test num if zero
                           0429
        0101'CF     D5     0429  20$:    TSTL    W^NUM                 ; Zero ?
             03     12     042D          BNEQ    30$                   ; No - generate addresses
           0046     31     042F          BRW     40$                   ; Yes - finish up this device
                           0432
    00000105'EF     DF     0432  30$:    PUSHAL  FX
    00000107'EF     DF     0438          PUSHAL  FL
   0549'CF   02     FB     043E          CALLS   #2,W^PUT_LINE         ; Output one line for this device
          03 50     E8     0443          BLBS    R0,35$                ; Branch if no overflow of addresses
           00D5     31     0446          BRW     EXIT                  ; Fatal error - exit
                           0449
                           0449  35$:    INC_CHAR                      ; Increment controller
                           0459
        0101'CF     D7     0459          DECL    W^NUM                 ; One less device
  50    011C'CF     9A     045D          MOVZBL  W^B_CNUMVEC,R0        ; Get number of vectors
       50    04     C4     0462          MULL2   #4,R0                 ; Calculate next vector addr.
       51    50     C0     0465          ADDL2   R0,R1                 ; Add in offset
50  0000011D'EF     3C     0468          MOVZWL  W_VECMOD,R0           ; Get vector modulus
       51    50     A0     046F          ADDW    R0,R1                 ; Round up to next vector
       51    50     AA     0472          BICW    R0,R1                 ; Truncate to actual vector offset
                           0475
           FFA9     31     0475          BRW     10$                   ; Loop
                           0478
        0101'CF     D5     0478  40$:    TSTL    W^NUM                 ; Better be zero
             03     13     047C          BEQL    50$                   ; If = 0 , o.k.
           0088     31     047E          BRW     TOO_MANY              ; If =/= 0 , error
                           0481
   0101'CF 6B47     9A     0481  50$:    MOVZBL  (R11)[R7],W^NUM       ; Has there been a change in R1
        0101'CF     D5     0487          TSTL    W^NUM                 ;   (other than rounding off)
             03     12     048B          BNEQ    60$                   ; Yes - restore w_vecbase
             FEF6   31  048D          BRW     LOOP                    ; No - loop
    0118'CF    51   B0  0490  60$:    MOVW    R1,W^W_VECBASE          ; Update vec_csrbase
             FEEE   31  0495          BRW     LOOP                    ; Loop
                        0498
                        0498
                        0498  FL_FL:  ;floating csr/floating vector
                        0498
    52  0114'CF     3C  0498  20$:    MOVZWL  W^W_CSRBASE,R2          ; R2 will be CSR register
         52    6A   A0  049D          ADDW    (R10),R2                ; Round to next csr
         52    6A   AA  04A0          BICW    (R10),R2                ; Truncate back to csr offset
                        04A3
          0101'CF   D5  04A3  30$:    TSTL    W^NUM                   ; Zero ?
               03   12  04A7          BNEQ    40$                     ; No - generate addresses
             004B   31  04A9          BRW     50$                     ; Restore vecbase, etc.
                        04AC
      00000107'EF   DF  04AC  40$:    PUSHAL  FL                      ; Floating
      00000107'EF   DF  04B2          PUSHAL  FL                      ; Floating
    0549'CF    02   FB  04B8          CALLS   #2,W^PUT_LINE           ; output one line for this device
            03 50   E8  04BD          BLBS    R0,45$                  ; Branch if no overflow of addresses
             005B   31  04C0          BRW     EXIT                    ; Fatal error - exit
                        04C3
                        04C3  45$:    INC_CHAR                        ; Increment controller
          0101'CF   D7  04D3          DECL    W^NUM                   ; One less device to do
    50  011C'CF     9A  04D7          MOVZBL  W^B_CNUMVEC,R0          ; Get number of controller int. vec.'s
         50    04   C4  04DC          MULL2   #4,R0                   ; Calculate next vector
         51    50   C0  04DF          ADDL2   R0,R1                   ; Add in offset
50    0000011D'EF   3C  04E2          MOVZWL  W_VECMOD,R0             ; Get vector modulus
         51    50   A0  04E9          ADDW    R0,R1                   ; Round up to next vector
         51    50   AA  04EC          BICW    R0,R1                   ; Truncate to actual vector offset
                        04EF
         52    6A   A0  04EF          ADDW    (R10),R2                ; Calculate next csr
               52   D6  04F2          INCL    R2                      ; Increment CSR
             FFAC   31  04F4          BRW     30$                     ; Loop
                        04F7
    0118'CF    51   B0  04F7  50$:    MOVW    R1,W^W_VECBASE          ; Save new vector offset
    0114'CF    52   B0  04FC          MOVW    R2,W^W_CSRBASE          ; Save new csr offset
                        0501
    0114'CF    02   A0  0501  60$:    ADDW    #2,W^W_CSRBASE          ; Advance past one register block
                        0506
             FE7D   31  0506          BRW     LOOP
                        0509
                        0509  TOO_MANY:
      007C8082 8F   DD  0509          PUSHL   #SYSG$_ABORT            ;  Fatal error - stop processing
               7E   D4  050F          CLRL    -(SP)
      007C80AA 8F   DD  0511          PUSHL   #SYSG$_TOO_MNY          ;  Illegal configuration
00000000'GF    03   FB  0517          CALLS   #3,G^LIB$SIGNAL         ;  Ran out of fixed addresses
                        051E
                        051E  EXIT:
               0D   E1  051E          BBC     #BOOCMD$V_OUTPUT,-
  0D 00000000'EF        0520                  BOO$GL_CMDOPT,10$       ; Branch if /OUTPUT not specified
                        0526          $CLOSE  FAB=RIO_OUTFAB2         ; Close files
                        0533  10$:
               0E   E1  0533          BBC     #BOOCMD$V_INPUT,-
  0D 00000000'EF        0535                  BOO$GL_CMDOPT,20$       ; Branch if /INPUT not specified
                        053B          $CLOSE  FAB=RIO_INFAB2          ; Close input file
                        0548  20$:
                    05  0548          RSB                             ; and return
```

```
0549   918
0549   919
```

```
                        0549          .SBTTL ROUTINE PUT_LINE
                        0549
                        0549   ;++
                        0549   ;
                        0549   ; routine to output csr and vector
                        0549   ;
                        0549   ;
                        0549   ; CALLING SEQUENCE
                        0549   ;
                        0549   ;       PUSHAL  <FX or FL>
                        0549   ;       PUSHAL  <FX or FL>
                        0549   ;       CALLS   #2,PUT_LINE
                        0549   ;
                        0549   ;
                        0549   ; INPUT
                        0549   ;
                        0549   ;  R1 is value of current vecbase (to be output)
                        0549   ;  R2 is value of current csrbase (ditto)
                        0549   ;  l_routine and l_devname give the current device name
                        0549   ;  R3 has the address of the flag byte for this device
                        0549   ;
                        0549   ;
                        0549   ; OUTPUT
                        0549   ;
                        0549   ; A single line of output for this device
                        0549   ;
                        0549   ;  (e.g. Device: DZ11   Name: TTA      CSR: ...  Vector ... Support ...)
                        0549   ;
                        0549   ;--
                        0549
                  0000  0549   .ENTRY  PUT_LINE,^M<>
                        054B
        7E    51    7D  054B            MOVQ    R1,-(SP)                ; Save R1,R2
51  00000200 8F     D1  054E            CMPL    #^01000,R1              ; Compare to largest legal value
              09    15  0555            BLEQ    10$                     ; Branch if erro
52  0000FFFF 8F     D1  0557            CMPL    #^0177777,R2            ; Compare to large-t legal value
              18    18  055E            BGEQ    20$                     ; Branch if OK
                        0560
                        0560   10$:
    007C8082 8F     DD  0560            PUSHL   #SYSG$_ABORT            ; Fatal error
              7E    D4  0566            CLRL    -(SP)
    007C8092 8F     DD  0568            PUSHL   #SYSG$_OVERFLOW         ; Overflow on addresses
 00000000'GF  03    FB  056E            CALLS   #3,G^LIB$SIGNAL         ; Signal error
              50    D4  0575            CLRL    R0                      ; Set error
                    04  0577            RET                             ; Return
                        0578
52  0003E000 8F     C8  0578   20$:     BISL2   #^0760000,R2            ; OR .n correct high order bits
0110'CF  010D'CF    9E  057F            MOVAB   W^NO,W^SUP              ; Initialize sup to "no"
         07 63  00  E1  0586            BBC     #UBA_V_SUPPORT,(R3),30$ ; Device supported ?
0110'CF  0109'CF    9E  058A            MOVAB   W^YES,W^SUP             ; Change sup to "yes"
                        0591
                        0591   30$:     $FAO_S  CTRSTR=DEV_LINE, -      ; Format string
                        0591                    OUTBUF=RIO$AB_OUTBUF,   -
                        0591                    OUTLEN=RIO$GW_OUTLEN,    -
                        0591                    P1=L_ROUTINE,     -     ; Routine name
                        0591                    P2=L_DEVNAME,     -     ; Device name
                        0591                    P3=R2,            -     ; Current csr
                  0591   978                  P4=8(AP),       -       ; '*' if floating, ' ' if fixed
                  0591   979                  P5=R1,          -       ; Current vector
                  0591   980                  P6=4(AP),       -       ; '*' if floating, ' ' if fixed
                  0591   981                  P7=SUP                  ; Yes/no for support
                  05C6   982
                  05C6   983          SIGNAL                          ; Signal if error
      FA34'   30  05C9   984          BSBW    RIO$OUTPUT_LINE         ; Output line
                  05CC   985
51      8E    7D  05CC   986 40$:     MOVQ    (SP)+,R1                ; Restore R1,R2
              04  05CF   987          RET                             ; Return
```

```
                             05D0  .SBTTL  ROUTINE REARNG_DEV
                             05D0
                             05D0  ;+
                             05D0  ;
                             05D0  ; FUNCTIONAL DESCRIPTION
                             05D0  ;
                             05D0  ;         This routine is used to rearrange devices in the DEVICES array.
                             05D0  ;         Its purpose is to move numbers from a device that changes from
                             05D0  ;         fixed,fixed allocation to fixed,floating or from fixed,floating
                             05D0  ;         to floating,floating. A number of devices on the UNIBUS are allocated
                             05D0  ;         I/O space in such a manner. CONFIGURE allows the input of these
                             05D0  ;         devices all in one line and this routine does the shuffling that
                             05D0  ;         is required to have one device at the first and the remanider
                             05D0  ;         at the second location in I/O space. Two calls to this routine must
                             05D0  ;         be made for DR11b's who have all three types of allocation.
                             05D0  ;
                             05D0  ; CALLING SEQUENCE
                             05D0  ;
                             05D0  ;         Via REARNG macro
                             05D0  ;
                             05D0  ; INPUT PARAMETERS
                             05D0  ;
                             05D0  ;         First(AP)  First name to lookup in UBATABLE
                             05D0  ;         Occ1(AP)   Which occurance of the first device to find
                             05D0  ;         Second(AP) Second name to lookup in UBATABLE
                             05D0  ;         Occ2(AP)   Which occurance of this device to find
                             05D0  ;         R11        Address of UBATABLE
                             05D0  ;
                             05D0  ; OUTPUT
                             05D0  ;
                             05D0  ;         As explained above and in the REARNG macro
                             05D0  ;
                             05D0  ;-
                             05D0
                             05D0  ; CONSTANTS
                             05D0
                   00000010  05D0  FIRST  = 16
                   0000000C  05D0  OCC1   = 12
                   00000008  05D0  SECOND = 8
                   00000004  05D0  OCC2   = 4
                             05D0
                       000C  05D0  .ENTRY  REARNG_DEV,^M<R2,R3>
                             05D2
  51    10 AC    D0  05D2          MOVL    FIRST(AP),R1           ; Address of ASCIC string
7E    51    01    C1  05D6          ADDL3   #1,R1,-(SP)            ; Push address of actual string
     7E    61    9A  05DA          MOVZBL  (R1),-(SP)             ; Push length of string
       0C AC    DD  05DD          PUSHL   OCC1(AP)               ; which occurance to find
FAFA CF    03    FB  05E0          CALLS   #3,LOOKUP              ; Find occurance in UBATABLE
52   00FD'CF    D0  05E5          MOVL    W^OFFSET,R2            ; Save position
                     05EA
  51    08 AC    D0  05EA          MOVL    SECOND(AP),R1          ; Address of ASCIC string
7E    51    01    C1  05EE          ADDL3   #1,R1,-(SP)            ; Push address of actual string
     7E    61    9A  05F2          MOVZBL  (R1),-(SP)             ; Push length of string
       04 AC    DD  05F5          PUSHL   OCC2(AP)               ; which occurance to find
FAE2 CF    03    FB  05F8          CALLS   #3,LOOKUP              ; Find occurance in UBATABLE
53   00FD'CF    D0  05FD          MOVL    W^OFFSET,R3            ; Save position
                     0602
```

```
                    6B43  95 0602  1046           TSTB    (R11)[R3]               ; Is DEVICES[second] = 0
                      1F  13 0605  1047           BEQL    10$                     ; Branch if OK
                             0607  1048
                             0607  1049 ;  Signal layout error
                             0607  1050
           007C8082 8F    DD 0607  1051           PUSHL   #SYSG$_ABORT            ; Unrecoverable - abort
                 10 AC    DD 060D  1052           PUSHL   FIRST(AP)               ; First device
                 08 AC    DD 0610  1053           PUSHL   SECOND(AP)              ; Second device
                    02    DD 0613  1054           PUSHL   #2                      ; Number of FAO argumants
           007C80A2 8F    DD 0615  1055           PUSHL   #SYSG$_INPUT_ERR        ; message name
     00000000'GF    05    FB 061B  1056           CALLS   #5,G^LIB$SIGNAL         ; Signal message
                    50    D4 0622  1057           CLRL    R0
                    15    11 0624  1058           BRB     30$
                             0626  1059
             6B42   01    91 0626  1060 10$:      CMPB    #1,(R11)[R2]            ; Is 1 >= DEVICES[first]
                    0C    18 062A  1061           BGEQ    20$                     ; If geq, ok
        6B43      6B42    90 062C  1062           MOVB    (R11)[R2],(R11)[R3]     ; Copy first to second
                  6B43    97 0631  1063           DECB    (R11)[R3]               ;  and subtract one from it
             6B42   01    90 0634  1064           MOVB    #1,(R11)[R2]            ; Force one of first type
                             0638  1065
               50   01    D0 0638  1066 20$:      MOVL    #1,R0
                          04 063B  1067 30$:      RET
                             063C  1068
```

```
                                     063C  1070 .SBTTL  VARIABLES USED IN SHOW/CONFIGURATION
                                     063C  1071
                             000002E7      1072 .PSECT PAGED_DATA       rd,wrt,noexe,quad
                                     02E7  1073 ;
                                     02E7  1074 ; variables used in SHOW/CONFIGURATION
                                     02E7  1075 ;
                                     02E7  1076
                                     02E7  1077 NAME:
                                     02E7  1078 NAME_L:
                           00000000  02E7  1079         .LONG   0                       ;DEVICE NAME DESCRIPTOR
                           000002EF' 02EB  1080         .LONG   NAME_S
58 58 58 58 58 58 58 58 58 58 58 58  02EF  1081 NAME_S: .ASCII  /XXXXXXXXXXXXXXX/
                           58 58 58  02FB
                           00000000  02FE  1082 UNIT:   .LONG   0                       ;NUMBER UNITS ON CONTROLLER
                           00000000  0302  1083 NVECT:  .LONG   0                       ;NUMBER OF DEVICE VECTORS ( 1 OR 2)
                           00000000  0306  1084 T_NVECT:.LONG   0                       ;TMP NVECT
                           00000000  030A  1085 AVECT1: .LONG   0                       ;IDB VECTOR 1 ADDRESS
                           00000000  030E  1086 AVECT2: .LONG   0                       ;IDB VECTOR 2 ADDRESS (IF PRESENT)
                           00000000  0312  1087 VCSR:   .LONG   0                       ;VIRTUAL CSR FOR DEVICE
                           00000000  0316  1088 CSR:    .LONG   0                       ;UNIBUS CSR ADDRESS
                                     031A  1089 COMBO_CSR:                              ;COMBO DEVICE'S CSR ADDRESS
                           00000000  031A  1090         .LONG   0                       ;
                                     031E  1091 COMBO_CSR_OFFSET:                       ;OFFSET TO START OF COMBO DEVICE'S CSRS
                           00000000  031E  1092         .LONG   0                       ;
                                     0322  1093 COMBO_VECTOR:                           ;COMBO DEVICE'S VECTOR ADDRESS
                           00000000  0322  1094         .LONG   0                       ;
                                     0326  1095 COMBO_VECTOR_OFFSET:                    ;OFFSET TO START OF COMBO DEVICE'S VECTORS
                           00000000  0326  1096         .LONG   0                       ;
                           00000000  032A  1097 TR:     .LONG   0                       ;SBI TR NUMBER
                                     032E  1098 ADP_TYPE:
                           00000000  032E  1099         .LONG   0                       ;ADDRESS OF ADP TEXT
                           00000000  0332  1100 OVECT1: .LONG   0                       ;VECTOR1
                           00000000  0336  1101 OVECT2: .LONG   0                       ;VECTOR2 (IF PRESENT)
                           00000000  033A  1102 UCB_SAVE: .LONG 0                       ;
                           00000000  033E  1103 DRIVER: .LONG   0
                                 00  0342  1104 DEV_FCUND: .BYTE 0                      ;
                           00000363  0343  1105 OTHER_BLOCK:    .BLKB   32
                                     0363  1106 SHOW_UBA:
3A 65 6D 61 4E 20 0000036B'010E0000' 0363  1107         .ASCID  @ Name: !4<!AS!> Units: !2<!UW!> Nexus:!2<!UW!> !5<(!AD)!>@ -
6E 55 20 3E 21 53 41 21 3C 34 21 20  0371
21 57 55 21 3C 32 21 20 3A 73 74 69  037D
21 3C 32 21 3A 73 75 78 65 4E 20 3E  0389
44 41 21 28 3C 35 21 20 3E 21 57 55  0395
                           3E 21 29  03A1
20 20 4C 4F 36 21 20 3A 52 53 43 20  03A4  1108                 @ CSR: !6OL  Vector1: !3OW  Vector2: !3OW@
4F 33 21 20 3A 31 72 6F 74 63 65 56  03B0
20 3A 32 72 6F 74 63 65 56 20 20 57  03BC
                        57 4F 33 21  03C8
                                     03CC  1109
                                     03CC  1110 SHOW_OTHER:
3A 65 6D 61 4E 20 000003D4'010E0000' 03CC  1111         .ASCID  @ Name: !4<!AS!> Units: !2<!UW!> Nexus:!2<!UW!> !5<(!AD)!> @
6E 55 20 3E 21 53 41 21 3C 34 21 20  03DA
21 57 55 21 3C 32 21 20 3A 73 74 69  03E6
21 3C 32 21 3A 73 75 78 65 4E 20 3E  03F2
44 41 21 28 3C 35 21 20 3E 21 57 55  03FE
                        20 3E 21 29  040A
                                     040E  1112
                                     040E  1113 CONNECT_UBA:
43 45 4E 4E 4F 43 00000416'010E0000' 040E  1114         .ASCID  @CONNECT !AS!UW /ADAP=!UW /CSR=%O!6OL /VECT=%O!3OW @ -
44 41 2F 20 57 55 21 53 41 21 20 54  041C
3D 52 53 43 2F 20 57 55 21 3D 50 41  0428
54 43 45 56 2F 20 4C 4F 36 21 4F 25  0434
            20 57 4F 33 21 4F 25 3D  0440
2F 20 57 4F 32 21 3D 56 4D 55 4E 2F  0448  1115                 @/NUMV=!2OW /DRIVER=!AC@
      43 41 21 3D 52 45 56 49 52 44  0454
                                     045E  1116 CONNECT_UBA2:
43 45 4E 4E 4F 43 00000466'010E0000' 045E  1117         .ASCID  @CONNECT !AS!UW /ADAP=!UW /CSR=%O!6OL /VECT=%O!3OW @ -
44 41 2F 20 57 55 21 53 41 21 20 54  046C
3D 52 53 43 2F 20 57 55 21 3D 50 41  0478
54 43 45 56 2F 20 4C 4F 36 21 4F 25  0484
            20 57 4F 33 21 4F 25 3D  0490
2F 20 57 4F 32 21 3D 56 4D 55 4E 2F  0498  1118                 @/NUMV=!2OW /DRIVER=!AC /CSR_OFFSET=!UB /VECTOR_OFFSET=!UB@
2F 20 43 41 21 3D 52 45 56 49 52 44  04A4
21 3D 54 45 53 46 46 4F 5F 52 53 43  04B0
4F 5F 52 4F 54 43 45 56 2F 20 42 55  04BC
         42 55 21 3D 54 45 53 46 46  04C8
                                     04D1  1119 CONNECT_OTHER:
4F 43 4F 54 55 41 000004D9'010E0000' 04D1  1120         .ASCID  @AUTOCONFIGURE !UW@
   57 55 21 20 45 52 55 47 49 46 4E  04DF
                                     04EA  1121
                                     04EA  1122 SHOCON_HEADER:
79 53 5F 21 2F 21 000004F2'010E0000' 04EA  1123         .ASCID  @!/!_System CSR and Vectors on !%D!/@
64 6E 61 20 52 53 43 20 6D 65 74 73  04F8
20 6E 6F 20 73 72 6F 74 63 65 56 20  0504
                     2F 21 44 25 21  0510
                                     0515  1124
                                     0515  1125 SAVE_HEADER:
20 4E 55 52 20 24 0000051D'010E0000' 0515  1126         .ASCID  @$ RUN SYS$SYSTEM:SYSGEN@
53 3A 4D 45 54 53 59 53 24 53 59 53  0523
                     4E 45 47 53 59  052F
                                     0534  1127
                                     0534  1128 ;
                                     0534  1129 ; The following is a table of longword text strings that are used
                                     0534  1130 ; to print adapter type. Proper ordering is assumed to follow
                                     0534  1131 ; the DCDEF definition.
                                     0534  1132 ;
                                     0534  1133 ; Note - There is a big assumption here that each ASCII string is
                                     0534  1134 ; EXACTLY four bytes long.
                                     0534  1135 ;
                                     0534  1136
                                     0534  1137 AL_ADP_TEXT:
                                     0534  1138
                                     0534  1139         Assume  AT$_MBA  EQ 0
                                     0534  1140         Assume  AT$_UBA  EQ 1
                                     0534  1141         Assume  AT$_DR   EQ 2
                                     0534  1142         Assume  AT$_MPM  EQ 3
                                     0534  1143         Assume  AT$_CI   EQ 4
                                     0534  1144         Assume  AT$_NULL EQ 5
                                     0534  1145
                         2041424D    0534  1146         .LONG   ^A/MBA /                ; adapter type = 0
                         20414255    0538  1147         .LONG   ^A/UBA /                ; adapter type = 1
                         20205244    053C  1148         .LONG   ^A/DR  /                ; adapter type = 2
                         204D504D    0540  1149         .LONG   ^A/MPM /                ; adapter type = 3
                         20204943    0544  1150         .LONG   ^A/CI  /                ; adapter type = 4
```

```
                        204C554E  0548   1151            .LONG   ^A/NUL /                 ; adapter type = 5
                        204B4E55  054C   1152            .LONG   ^A/UNK /                 ; adapter type >= 6 - UNKNOWN
                                  0550   1153
                        00000007  0550   1154 L_MAXADP:          .LONG   7                ; Highest legal adapter type + 1
```

```
                                  0554 .SBTTL  INITIALIZATION CODE FOR SYSTEM DUMP
                                  0554
                                  0554 ;
                                  0554 ;SHOW/CONFIGURATION - Rick Spitz
                                  0554 ;SHOW/CON/COMMAND
                                  0554 ;
                                  0554 ;        THIS PROGRAM WILL DUMP OUT A LIST OF CSR AND VECTOR
                                  0554 ;        ADDRESSES CONTAINED IN THE VMS DEVICE DATA STRUCTURES
                                  0554 ;
                                  0554 ;
                                  0554
                              0000063C .PSECT  PAGED_CODE      rd,nowrt,exe,long
                                  063C
                           OFFC   063C .ENTRY  BOO$SHOCONFIG, ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                  063E
      52    00000343'EF    DE     063E         MOVAL   OTHER_BLOCK,R2              ; Zero other block
                     82    7C     0645         CLRQ    (R2)+                       ; 8 bytes at a time
                     82    7C     0647         CLRQ    (R2)+                       ; (this assumes max of 32 nexuses)
                     82    7C     0649         CLRQ    (R2)+
                     82    7C     064B         CLRQ    (R2)+
                                  064D
                  F9B0'    30     064D 5$:     BSBW    BOO$OPEN_OUTPUT_2           ; Open /output= file (D=sys$output)
                  06 50    E8     0650         BLBS    R0,20$                      ; Branch if ok
                                  0653
                                  0653         SIGNAL                              ; Signal error
                   0073    31     0656         BRW     40$                         ; exit on error
                                  0659
  1B 00000000'EF     02    E0     0659 20$:    BBS     #BOOCMD$V_SAVE,BOO$GL_CMDOPT,25$ ; Branch if SAVE command
                                  0661         $FAO_S  CTRSTR=SHOCON_HEADER, -
                                  0661                 OUTBUF=RIO$AB_OUTBUF,-
                                  0661                 OUTLEN=RIO$GW_OUTLEN        ; Format string
                     19    11     067A         BRB     30$
                                  067C
                                  067C 25$:    $FAO_S  CTRSTR=SAVE_HEADER, -
                                  067C                 OUTBUF=RIO$AB_OUTBUF,-
                                  067C                 OUTLEN=RIO$GW_OUTLEN        ; Format string
                                  0695
                                  0695 30$:    SIGNAL                              ; Signal if error
                  F965'    30     0698         BSBW    RIO$OUTPUT_LINE
                                  069B
                                  069B         $CMEXEC_S       EXEC                ; Change mode to exec to allow
                                  06AA                                             ;  read to executive data structures
                                  06AA         SIGNAL                              ; Check for error
                  1C 50    E9     06AD         BLBC    R0,40$                      ; Branch on error
         15 00000342'EF    E8     06B0         BLBS    DEV_FOUND,40$               ; Branch if device has been printed
            000001BF'EF    DD     06B7         PUSHL   BOO$GL_TR                   ; Adapter #
                     01    DD     06BD         PUSHL   #1                          ; # of FAO params
            007C9020 8F    DD     06BF         PUSHL   #SYSG$_NODEVADAP            ; Set warning "no devices on adapter"
    00000000'GF      03    FB     06C5         CALLS   #3,G^LIB$SIGNAL             ; Signal error
                                  06CC 40$:
                     0D    E1     06CC         BBC     #BOOCMD$V_OUTPUT,-
       0D 00000000'EF             06CE                 BOO$GL_CMDOPT,50$           ; Branch if /OUTPUT not specified
                                  06D4         $CLOSE  FAB=RIO_OUTFAB2             ; Close file
    000001BF'EF      01    CE     06E1 50$:    MNEGL   #1,BOO$GL_TR                ; Clear TR for subsequent calls
               50    01    D0     06E8         MOVL    #1,R0                       ; Set success for tparse
                           04     06EB         RET                                 ; Return
                                  06EC
```

```
                                   06EC   1214 .SBTTL  DATA BASE SCAN
                                   06EC   1215
                                   06EC   1216 ;       EXEC MODE ROUTINE TO SCAN DEVICE DATA BASE AND
                                   06EC   1217 ;       COMPUTE CSR AND VECTORS FOR EACH DEVICE IN THE SYSTEM
                                   06EC   1218 ;
                                   06EC   1219 ;       REGISTER USAGE:
                                   06EC   1220 ;               R2      DDB
                                   06EC   1221 ;               R3      UCB
                                   06EC   1222 ;               R4      CRB
                                   06EC   1223 ;               R5      ADP
                                   06EC   1224 ;
                                   06EC   1225
                            000C   06EC   1226 .ENTRY  EXEC, ^M<R2,R3>                       ; Entry mask
            00000342'EF       94   06EE   1227         CLRB    DEV_FOUND                     ; Zero flag
      52    00000000'GF       D0   06F4   1228         MOVL    G^IOC$GL_DEVLIST,R2           ; DDB header
                      08      11   06FB   1229         BRB     DDB1
                                   06FD   1230
              52      62      D0   06FD   1231 DDBLOOP: MOVL   DDB$L_LINK(R2),R2             ; Get next ddb address
                      03      12   0700   1232         BNEQ    DDB1                          ; More to do
                    038B      31   0702   1233         BRW     DONE                          ; Finished
                                   0705   1234
        000002FE'EF   01      9A   0705   1235 DDB1:   MOVZBL  #1,UNIT                       ; Clear unit count
            00000332'EF       D4   070C   1236         CLRL    OVECT1                        ; Output vectors
            00000336'EF       D4   0712   1237         CLRL    OVECT2                        ; 1 & 2
            53     14 A2      9E   0718   1238         MOVAB   DDB$T_NAME(R2),R3             ; Address of generic name
        000002E7'EF   83      9A   071C   1239         MOVZBL  (R3)+,NAME_L                  ; Length of name
                                   0723   1240
                      3F      BB   0723   1241         PUSHR   #^M<R0,R1,R2,R3,R4,R5>  ; save for move character
 0F    20    63   000002E7'EF 2C   0725   1242         MOVC5   NAME_L,(R3),#SPACE,#15,NAME_S   ; copy name
            000002EF'EF            072E
                      3F      BA   0733   1243         POPR    #^M<R0,R1,R2,R3,R4,R5>
                                   0735   1244
            53     04 A2      D0   0735   1245         MOVL    DDB$L_UCB(R2),R3              ; address of first ucb
                      C2      13   0739   1246         BEQL    DDBLOOP                       ; no UCB's there
        0000033A'EF   53      D0   073B   1247         MOVL    R3,UCB_SAVE                   ; save for SAVE command
            54     24 A3      D0   0742   1248         MOVL    UCB$L_CRB(R3),R4              ; CRB address
                                   0746   1249
            53     30 A3      D0   0746   1250 UCB1:   MOVL    UCB$L_LINK(R3),R3             ; Follow UCB link
                      08      13   074A   1251         BEQL    10$                           ; no more units,try next device
            000002FE'EF       D6   074C   1252         INCL    UNIT                          ; Increment units on this controller
                      F2      11   0752   1253         BRB     UCB1                          ; Skip this unit
                                   0754   1254
            00000302'EF       D4   0754   1255 10$:    CLRL    NVECT                         ; Init number of vectors
            0000030A'EF       7C   075A   1256         CLRQ    AVECT1                        ; Init both vectors
        006C 8F    08 A4      B1   0760   1257         CMPW    CRB$W_SIZE(R4),#CRB$C_LENGTH+VEC$C_LENGTH ; Single vector?
                      0E      19   0766   1258         BLSS    VECT1                         ; 1 vector
                                   0768   1259 ;
                                   0768   1260 ;       DEVICE HAS TWO VECTORS
                                   0768   1261 ;
      0000030E'EF    4A A4    DE   0768   1262         MOVAL   CRB$L_INTD2+2(R4),AVECT2 ; Address of second vector
            00000302'EF       D6   0770   1263         INCL    NVECT                         ; Count of vectors
                                   0776   1264 VECT1:
      0000030A'EF    26 A4    DE   0776   1265         MOVAL   CRB$L_INTD+2(R4),AVECT1
            00000302'EF       D6   077E   1266         INCL    NVECT
 00000306'EF    00000302'EF   D0   0784   1267         MOVL    NVECT,T_NVECT                 ; Save count in temp
            55     2C A4      D0   078F   1268         MOVL    CRB$L_INTD+VEC$L_IDB(R4),R5 ; Idb address
        00000312'EF   65      D0   0793   1269         MOVL    IDB$L_CSR(R5),VCSR            ; Virtual csr addresss
```

```
                  0F A5   98  079A  1270         CVTBL   IDB$B_COMBO_CSR_OFFSET(R5),-;Get offset to start of combo
           0000031E'EF        079D  1271                 COMBO_CSR_OFFSET          ; device's CSRs
                  10 A5   98  07A2  1272         CVTBL   IDB$B_COMBO_VECTOR_OFFSET(R5),-;Get offset to start of combo
           00000326'EF        07A5  1273                 COMBO_VECTOR_OFFSET       ; device's VECTORs
                  10 A5   83  07AA  1274         SUBB3   IDB$B_COMBO_VECTOR_OFFSET(R5),-;Get vector address(in longwords)
   00000322'EF    0B A5       07AD  1275                 IDB$B_VECTOR(R5),COMBO_VECTOR; start of combo device's vectors
           55     14 A5   D0  07B4  1276         MOVL    IDB$L_ADP(R5),R5          ; Adaptor block for device
                     03   12  07B8  1277         BNEQ    20$                       ; Physical device
                   FF40   31  07BA  1278         BRW     DDBLOOP                   ; Skip it
                              07BD  1279
   0000032A'EF    0C A5   3C  07BD  1280 20$:    MOVZWL  ADP$W_TR(R5),TR           ; TR number for adaptor
        57   000001BF'EF  D0  07C5  1281         MOVL    BOO$GL_TR,R7              ; Input from /ADAPTER =
                     0C   19  07CC  1282         BLSS    22$                       ; Branch if non-existent
        57   0000032A'EF  D1  07CE  1283         CMPL    TR,R7                     ; Must be equal
                     03   13  07D5  1284         BEQL    22$                       ; Branch if yes
                   FF23   31  07D7  1285         BRW     DDBLOOP                   ; Try another device
                              07DA  1286
                              07DA  1287 ;
                              07DA  1288 ; Get text string associated with adapter number
                              07DA  1289 ;
                              07DA  1290
              57  0E A5   3C  07DA  1291 22$:    MOVZWL  ADP$W_ADPTYPE(R5),R7      ; Get adapter type
        57   00000550'8F  D1  07DE  1292         CMPL    #L_MAXADP,R7              ; Compare to legal maximum
                     07   18  07E5  1293         BGEQ    25$                       ; Branch if legal
        57   00000550'8F  D0  07E7  1294         MOVL    #L_MAXADP,R7              ; Set to unknown (UNK)
                              07EE  1295
                 57  04   C4  07EE  1296 25$:    MULL    #4,R7                     ; Set up for longword offset
        56   00000534'EF  DE  07F1  1297         MOVAL   AL_ADP_TEXT,R6            ; Get address of text table
   0000032E'EF    56  57  C1  07F8  1298         ADDL3   R7,R6,ADP_TYPE            ; Add them together
                              0800  1299
              0E A5  01   B1  0800  1300         CMPW    #AT$_UBA,ADP$W_ADPTYPE(R5) ; Is it a UBA?
                     03   13  0804  1301         BEQL    30$                       ; yes
                   007B   31  0806  1302         BRW     PRINT                     ; no,print (no device vectors)
                              0809  1303 30$:
              56  10 A5   D0  0809  1304         MOVL    ADP$L_VECTOR(R5),R6       ; vector table address
                              080D  1305 ;
                              080D  1306         .SBTTL  VECTOR TABLE SCAN
                              080D  1307
                              080D  1308 ;       SCAN VECTOR TABLE FOR PROPER VECTOR
                              080D  1309
                     57   D4  080D  1310         CLRL    R7                        ; init count
                              080F  1311 100$:
                 58  86   D0  080F  1312         MOVL    (R6)+,R8                  ; get a vector
           0000030A'EF 58 D1  0812  1313         CMPL    R8,AVECT1                 ; is it first
                     39   13  0819  1314         BEQLU   130$                      ; yes
           0000030E'EF 58 D1  081B  1315         CMPL    R8,AVECT2                 ; second?
                     30   13  0822  1316         BEQLU   130$                      ; yes
                 58  01   CA  0824  1317         BICL    #1,R8                     ; mask interrupt stack bit
                 58  02   C0  0827  1318         ADDL2   #2,R8                     ; 11/750 vectors point to pushr
           0000030A'EF 58 D1  082A  1319         CMPL    R8,AVECT1                 ; is it first
                     21   13  0831  1320         BEQLU   130$                      ; yes
           0000030E'EF 58 D1  0833  1321         CMPL    R8,AVECT2                 ; second?
                     18   13  083A  1322         BEQLU   130$                      ; yes
                              083C  1323 110$:
   CB 57   00000080 8F    F2  083C  1324         AOBLSS  #128,R7,100$              ; try next one
           00000332'EF 01 CE  0844  1325         MNEGL   #1,OVECT1                 ; error no vector found (or only 1 of 2)
           00000336'EF 01 CE  084B  1326         MNEGL   #1,OVECT2
```

```
                                   30   11  0852          BRB     PRINT
                                            0854  130$:
                      00000306'EF   D7  0854          DECL    T_NVECT                  ; first or second?
                               0D   18  085A          BGEQ    140$                     ; first
                     58   57   02   78  085C          ASHL    #2,R7,R8                 ; mult by 4 ( vectors are 4 bytes long)
                 00000336'EF   58   D0  0860          MOVL    R8,OVECT2
                               1B   11  0867          BRB     PRINT                    ; both vectors loaded,print
                                            0869  140$:
                     58   57   02   78  0869          ASHL    #2,R7,R8                 ; mult by 4
                 00000332'EF   58   D0  086D          MOVL    R8,OVECT1
                      00000306'EF   D5  0874          TSTL    T_NVECT                  ; if single vector, then done
                               08   13  087A          BEQL    PRINT
                      00000306'EF   D4  087C          CLRL    T_NVECT                  ; reset so second vector goes to
                                            0882                                       ; to field 2
                               B8   11  0882          BRB     110$
                                            0884  ;
                                            0884          .SBTTL  VIRTUAL CSR CONVERSION
                                            0884
                                            0884  ;       CONVERT CSR TO PHYSICAL AND PRINT RECORD
                                            0884  ;
                                            0884  PRINT:
                 50   00000000'GF   D0  0884          MOVL    G^MMG$GL_SPTBASE,R0      ; System page table
                 51   00000312'EF   D0  088B          MOVL    VCSR,R1                  ; Virtual CSR
               51   51   15   09   EF  0892          EXTZV   #9,#21,R1,R1             ; Get VPN
                          50   6041   D0  0897          MOVL    (R0)[R1],R0              ; Get PTE
                     50   50   09   78  089B          ASHL    #9,R0,R0                 ; Shift to page field
      50    09   00   00000312'EF   F0  089F          INSV    VCSR,#0,#9,R0            ; Add byte offset
                 50   FFFC0000 8F   CA  08A8          BICL2   #^XFFFC0000,R0           ; Mask all but low 18 bits
                 50   00026200 8F   D1  08AF          CMPL    #^O461000,R0             ; RB730 CSR in adapter space?
                               07   12  08B6          BNEQ    3$                       ; If neq no
                 50   0003FB86 8F   D0  08B8          MOVL    #^O775606,R0             ; Set UNIBUS space CSR address
                 50   0003E000 8F   C8  08BF  3$:     BISL2   #^O760000,R0             ; OR in correct high order bits
                 00000316'EF   50   D0  08C6          MOVL    R0,CSR                   ; Save CSR address
0000031A'EF      50   0000031E'EF   C1  08CD          ADDL3   COMBO_CSR_OFFSET,R0,-    ; Calculate CSR of combo device
                                            08D9                  COMBO_CSR                ;
                      0000031E'EF   CE  08D9          MNEGL   COMBO_CSR_OFFSET,-       ; Make offset from combo device CSR start
                      0000031E'EF       08DF                  COMBO_CSR_OFFSET         ;
00000326'EF      00000326'EF   02   78  08E4          ASHL    #2,COMBO_VECTOR_OFFSET,COMBO_VECTOR_OFFSET; Get offset in bytes
00000322'EF      00000322'EF   02   78  08F0          ASHL    #2,COMBO_VECTOR,COMBO_VECTOR; Get combo vector address in bytes
                               0B   12  08FC          BNEQ    5$                       ; Branch if not the UNIBUS boot device
                                            08FE                                       ; (IDB$B_COMBO_VECTOR not currently filled i
      00000322'EF      00000332'EF   D0  08FE          MOVL    OVECT1,COMBO_VECTOR      ; Assume boot device is not part of a combo
                                            0909  5$:
                                            0909          .SBTTL  MESSAGE OUTPUT
                                            0909  ;
                                            0909  ;       BUILD MESSAGE
                                            0909  ;
                 00000342'EF   01   90  0909          MOVB    #1,DEV_FOUND             ; Set device found flag
              03 00000000'EF   02   E1  0910          BBC     #BOOCMD$V_SAVE,BOO$GL_CMDOPT,10$ ; Branch if SHOW/CONFIG command
                             0089   31  0918          BRW     SAVE_OUTPUT              ; Branch to SAVE output
                                            091B
                        0E A5   01   B1  091B  10$:    CMPW    #AT$_UBA,ADP$W_ADPTYPE(R5) ; Is it a UBA?
                               47   12  091F          BNEQ    20$                      ; Branch if not
                                            0921          $FAO_S  CTRSTR=SHOW_UBA,-
                                            0921                  OUTBUF=RIO$AB_OUTBUF,-
                                            0921                  OUTLEN=RIO$GW_OUTLEN,-
                                            0921                  P1=#NAME, -
                          0921  1384                    P2=UNIT,  -
                          0921  1385                    P3=TR,    -
                          0921  1386                    P4=#3,    -
                          0921  1387                    P5=ADP_TYPE,-
                          0921  1388                    P6=CSR,   -
                          0921  1389                    P7=OVECT1,-
                          0921  1390                    P8=OVECT2
                  33  11  0966  1391         BRB        30$
                          0968  1392
                          0968  1393 20$:    $FAO_S     CTRSTR=SHOW_OTHER,-
                          0968  1394                    OUTBUF=RIO$AB_OUTBUF,-
                          0968  1395                    OUTLEN=RIO$GW_OUTLEN,-
                          0968  1396                    P1=#NAME, -
                          0968  1397                    P2=UNIT,  -
                          0968  1398                    P3=TR,    -
                          0968  1399                    P4=#3,    -
                          0968  1400                    P5=ADP_TYPE
                          099B  1401 ;
                          099B  1402 ;       OUTPUT MESSAGE
                          099B  1403 ;
                          099B  1404 30$:    SIGNAL                          ; FAO error?
               F65F'  30  099E  1405         BSBW       RIO$OUTPUT_LINE      ; Output line
                FD59  31  09A1  1406         BRW        DDBLOOP              ; Get next device
                          09A4  1407
                          09A4  1408 SAVE_OUTPUT:
      53  0000033A'EF  D0  09A4  1409         MOVL       UCB_SAVE,R3          ; Head of list
 0000033E'EF    24 A2  DE  09AB  1410         MOVAL      DDB$T_DRVNAME(R2),DRIVER ; Driver name
         56     54 A3  D0  09B3  1411 20$:    MOVL       UCB$W_UNIT(R3),R6    ; Unit number
                          09B7  1412
            0E A5   01  B1  09B7  1413         CMPW       #AT$_UBA,ADP$W_ADPTYPE(R5) ; Is it a UBA?
                    03  13  09BB  1414         BEQL       23$                  ; EQL yes
                  0096  31  09BD  1415         BRW        40$                  ; No, other
                          09C0  1416
           0000031E'EF  D5  09C0  1417 23$:    TSTL       COMBO_CSR_OFFSET     ; Is this a combo device?
                    41  12  09C6  1418         BNEQ       25$                  ; If neq yes
                          09C8  1419         $FAO_S     CTRSTR=CONNECT_UBA,- ; Format CONNECT line
                          09C8  1420                    OUTBUF=RIO$AB_OUTBUF,-
                          09C8  1421                    OUTLEN=RIO$GW_OUTLEN,-
                          09C8  1422                    P1=#NAME, -
                          09C8  1423                    P2=R6,    -
                          09C8  1424                    P3=TR,    -
                          09C8  1425                    P4=CSR,   -
                          09C8  1426                    P5=OVECT1,-
                          09C8  1427                    P6=NVECT, -
                          09C8  1428                    P7=DRIVER
                  75  11  0A07  1429         BRB        50$
                          0A09  1430
                          0A09  1431 25$:    $FAO_S     CTRSTR=CONNECT_UBA2,- ; Format CONNECT line
                          0A09  1432                    OUTBUF=RIO$AB_OUTBUF,-
                          0A09  1433                    OUTLEN=RIO$GW_OUTLEN,-
                          0A09  1434                    P1=#NAME, -
                          0A09  1435                    P2=R6,    -
                          0A09  1436                    P3=TR,    -
                          0A09  1437                    P4=COMBO_CSR,-
                          0A09  1438                    P5=COMBO_VECTOR,-
                          0A09  1439                    P6=NVECT, -
                          0A09  1440                    P7=DRIVER,-
```

```
                               0A09                 P8=COMBO_CSR_OFFSET,-
                               0A09                 P9=COMBO_VECTOR_OFFSET
                  28     11    0A54         BRB     50$
                               0A56
        50    0C A5      3C    0A56  40$:   MOVZWL  ADP$W_TR(R5),R0         ; Get nexus number
2B 00000343'EF    50     E2    0A5A         BBSS    R0,OTHER_BLOCK,80$      ; Ignore if this one's been done
                               0A62         $FAO_S  CTRSTR=CONNECT_OTHER,-  ; Format AUTOCONFIGURE command for bus
                               0A62                 OUTBUF=RIO$AB_OUTBUF,-
                               0A62                 OUTLEN=RIO$GW_OUTLEN,-
                               0A62                 P1=ADP$W_TR(R5)
                               0A7E
                               0A7E  ;
                               0A7E  ;              OUTPUT CONNECT LINE
                               0A7E  ;
                               0A7E  50$:   SIGNAL                          ; Check FAO status
               F57C'     30    0A81         BSBW    RIO$OUTPUT_LINE         ; Output the line
                               0A84
        53    30 A3      D0    0A84  70$:   MOVL    UCB$L_LINK(R3),R3       ; Next UCB
                  03     13    0A88         BEQL    80$                     ; Branch if at end of list
                FF26     31    0A8A         BRW     20$                     ; Loop if not
                               0A8D
                FC6D     31    0A8D  80$:   BRW     DDBLOOP                 ; Get next device
                               0A90
             50   01     D0    0A90  DONE:  MOVL    #1,R0                   ; Set success
                         04    0A93         RET                             ; Return
                               0A94
```

```
                            0A94  .SBTTL  BOO$SHOW_UNIBUS
                            0A94
                            0A94  ;++
                            0A94  ; BOO$SHOW_UNIBUS
                            0A94  ;
                            0A94  ; FUNCTIONAL DESCRIPTION:
                            0A94  ;
                            0A94  ; This routine is called via the SHOW/UNIBUS command. It's function is
                            0A94  ; to test every word of memory in UNIBUS I/O space and return the
                            0A94  ; CSR, virtual address of that CSR, and the data at that location if
                            0A94  ; the address responds. This is used as a debugging aid for new UNIBUS
                            0A94  ; configurations. Command requires CMKRNL and results in IPL being raised.
                            0A94  ;
                            0A94  ; NOTE that reading a CSR may remove a character from a buffer or otherwise
                            0A94  ; make the device act strange.
                            0A94  ;
                            0A94  ;
                            0A94  ;
                            0A94  ; CALLING SEQUENCE:
                            0A94  ;
                            0A94  ;       Called as a TPARSE action routine.
                            0A94  ;
                            0A94  ; INPUT PARAMETERS:
                            0A94  ;
                            0A94  ;       TPA$L_PARAM(AP) = 0, if no /ADAPTER was specified.
                            0A94  ;                       = 1, if /ADAPTER was specified.
                            0A94  ;       BOO$GL_TR has nexus number if TPA$L_PARAM(AP) = 1
                            0A94  ;
                            0A94  ; IMPLICIT INPUTS:
                            0A94  ;       NONE
                            0A94  ;
                            0A94  ; OUTPUT PARAMETERS:
                            0A94  ;       NONE
                            0A94  ;
                            0A94  ; IMPLICIT OUTPUTS:
                            0A94  ;       NONE
                            0A94  ;
                            0A94  ; COMPLETION CODES:
                            0A94  ;
                            0A94  ;       SS$_NORMAL
                            0A94  ;       SS$_NOPRIV - No privilege
                            0A94  ;       SYSG$_NEXNOTUBA - nexus not a UNIBUS
                            0A94  ;
                            0A94  ; SIDE EFFECTS:
                            0A94  ;
                            0A94  ;       All UNIBUS I/O locations that respond are read from.
                            0A94  ;
                            0A94  ;--
                            0A94
                      01FC  0A94  .Entry  BOO$SHOW_UNIBUS,^M<R2,R3,R4,R5,R6,R7,R8>
                            0A96
         30 20 AC     E8    0A96          BLBS    TPA$L_PARAM(AP),100$    ; Branch if /ADAPTER specified
                            0A9A  ;
                            0A9A  ; Loop through all UNIBUS's on system
                            0A9A  ;
                            0A9A
57    00000000'GF     D0    0A9A          MOVL    G^EXE$GL_NUMNEXUS,R7    ; Number of nexuses
```

```
                 58  D4  0AA1        CLRL    R8                        ; Count
                         0AA3
                         0AA3  10$:  $CMEXEC_S W^GET_ADP               ; Loop through all nexuses
  50   007C80E2 8F   D1  0AB0        CMPL    #SYSG$_NEXNOTUBA,R0       ; nexus not UNIBUS?
                 0B  13  0AB7        BEQL    20$                       ; Error expected, continued
              2E 50  E9  0AB9        BLBC    R0,210$                   ; Other error, exit
                         0ABC
        0AF2'CF  00  FB  0ABC        CALLS   #0,W^One_Unibus           ; UNIBUS found, format
              26 50  E9  0AC1        BLBC    R0,210$                   ; Exit on error
                         0AC4
         DB 58   57  F2  0AC4  20$:  AOBLSS  R7,R8,10$                 ; LOOP
                 1D  11  0AC8        BRB     200$                      ; Exit, status OK
                         0ACA
                         0ACA  ;
                         0ACA  ; Do a single adapter as specified
                         0ACA  ;
                         0ACA
                         0ACA  100$:
     58  01BF'CF     D0  0ACA        MOVL    W^BOO$GL_TR,R8            ; Set nexus number
                         0ACF        $CMEXEC_S W^GET_ADP               ; Is it a UNIBUS ?, get CSR
              0B 50  E9  0ADC        BLBC    R0,210$                   ; Branch on any error
                         0ADF
        0AF2'CF  00  FB  0ADF        CALLS   #0,W^One_Unibus           ; UNIBUS found, format
              03 50  E9  0AE4        BLBC    R0,210$                   ; Exit on error
                         0AE7
              50 01  D0  0AE7  200$: MOVL    #SS$_NORMAL,R0            ; Set success
    000001BF'EF  01  CE  0AEA  210$: MNEGL   #1,BOO$GL_TR              ; Default TR for subsequent calls
                     04  0AF1        RET                               ; Return to TPARSE
                         0AF2
                         0AF2
```

```
                                0AF2  ;+
                                0AF2  ; One_Unibus
                                0AF2  ;
                                0AF2  ; FUNCTIONAL DESCRIPTION:
                                0AF2  ;
                                0AF2  ; Format the data for a single UNIBUS.
                                0AF2  ;
                                0AF2  ; CALLING SEQUENCE:
                                0AF2  ;
                                0AF2  ;         Calls   #0,One_Unibus
                                0AF2  ;
                                0AF2  ; INPUT PARAMETERS:
                                0AF2  ;
                                0AF2  ;         R6 = ADP CSR
                                0AF2  ;         R8 = Adapter number
                                0AF2  ;
                                0AF2  ;--
                                0AF2
                          0000  0AF2  One_Unibus: .word ^M<>
                                0AF4
                                0AF4            $FAO_S  CTRSTR = FAO_Q_ONEUBA,-
                                0AF4                    OUTBUF = RIO$AB_OUTBUF,-
                                0AF4                    OUTLEN = RIO$GW_OUTLEN,-
                                0AF4                    P1 = R8
                    F4EE'   30  0B0F            BSBW    RIO$OUTPUT_LINE
                                0B12
                       53   D4  0B12            CLRL    R3                      ; To be offset into UBA I/O space
                                0B14
                                0B14  10$:      $CMKRNL_S L^CHECK_CSR
        50   007C9040 8F    D1  0B23            CMPL    #SYSG$_ENDUBA,R0        ; End of UNIBUS I/O Space?
                       49   13  0B2A            BEQL    100$                    ; Branch if done
                    49 50   E9  0B2C            BLBC    R0,200$                 ; Exit if error
                                0B2F
        50   007CA02B 8F    D1  0B2F            CMPL    #SYSG$_SKIPPED,R0       ; Was section skipped?
                       0E   12  0B36            BNEQ    20$                     ; Branch if not
                                0B38
                       51   DD  0B38            PUSHL   R1                      ; Save value in CSR
             00000000'EF    B4  0B3A            CLRW    RIO$GW_OUTLEN           ; Zero length
                    F4BD'   30  0B40            BSBW    RIO$OUTPUT_LINE         ; Output a blank line
                     51 8ED0    0B43            POPL    R1                      ; Restore value in CSR
                                0B46
                                0B46  20$:
   55    53   0003E000 8F   C9  0B46            BISL3   #^O760000,R3,R5         ; Octal offset
                                0B4E            $FAO_S  CTRSTR = FAO_D_CTRSTR,-
                                0B4E                    OUTBUF = RIO$AB_OUTBUF,-
                                0B4E                    OUTLEN = RIO$GW_OUTLEN,-
                                0B4E                    P1 = R5,-
                                0B4E                    P2 = R2,-
                                0B4E                    P3 = R1
                                0B6D
                    08 50   E9  0B6D            BLBC    R0,200$                 ; Exit on error
                    F48D'   30  0B70            BSBW    RIO$OUTPUT_LINE         ; Output line
                       9F   11  0B73            BRB     10$                     ; Loop
                                0B75
                  50   01   D0  0B75  100$:     MOVL    #SS$_NORMAL,R0          ; Set success
                            04  0B78  200$:     RET                             ; Exit
                                0B79
```

```
                                  0B79  ;+
                                  0B79  ;       CHECK_CSR
                                  0B79  ;
                                  0B79  ; CALLING SEQUENCE:
                                  0B79  ;
                                  0B79  ;       $CMKRNL CHECK_CSR
                                  0B79  ;
                                  0B79  ; INPUT:
                                  0B79  ;
                                  0B79  ;       R3 = CSR address offset
                                  0B79  ;       R6 = address of ADP CSR for that nexus
                                  0B79  ;
                                  0B79  ; OUTPUT:
                                  0B79  ;
                                  0B79  ;       R1 = Data at location
                                  0B79  ;       R2 = CSR virtual address
                                  0B79  ;       R3 = CSR offset (Context)
                                  0B79  ;       R6 = unchanged
                                  0B79  ;
                                  0B79  ; RETURN STATUS:
                                  0B79  ;
                                  0B79  ;       SS$_NORMAL     - Next CSR responded
                                  0B79  ;       SYSG$_SKIPPED - responding CSR was found but only after skipping
                                  0B79  ;                       at least one word in the UNIBUS I/O Space.
                                  0B79  ;       SYSG$_ENDUBA  - End of UNIBUS I/O Space was encountered.
                                  0B79  ;
                                  0B79  ;
                                  0B79  ;-
                                  0B79
                              00000000          .PSECT  NONPAGED_CODE   rd,nowrt,exe,long
                                  0000
                            0000  0000  CHECK_CSR:      .word   0                       ; Entry mask
                                  0002
                         01   DD  0002          PUSHL   #SS$_NORMAL                     ; Assume success
                                  0004
50   007C9040 8F              D0  0004  10$:    MOVL    #SYSG$_ENDUBA,R0                ; Assume end
53   00001FFE 8F              D1  000B          CMPL    #^017776,R3                     ; Loop through legal CSR's
                         30   15  0012          BLEQ    40$                             ; Exit if done
                                  0014
                    53   02   C0  0014          ADDL2   #2,R3                           ; Add 2 and try again
     52   1000 C643           9E  0017          MOVAB   UBA_IOBASE(R6)[R3],R2           ; Calcuate UNIBUS CSR address
                                  001D
                    50   52   D0  001D          MOVL    R2,R0                           ; Set input for EXE$TEST_CSR (R6 too.)
                                  0020          DSBINT                                  ; Disable interrupts  (needed for 780)
     00000000'GF              16  0026          JSB     G^EXE$TEST_CSR                  ; Test location
                    0C   50   E8  002C          BLBS    R0,30$                          ; LBS if location exists
                                  002F          ENBINT                                  ; Restore IPL
6E   007CA02B 8F              D0  0032          MOVL    #SYSG$_SKIPPED,(SP)             ; Set section skipped
                         C9   11  0039          BRB     10$                             ; Loop
                                  003B
                    51   62   3C  003B  30$:    MOVZWL  (R2),R1                         ; Get data in register, Zero high word
                                  003E          ENBINT                                  ; Enable interrupts
                         50 8ED0  0041          POPL    R0                              ; Set success
                                  0044
                              04  0044  40$:    RET
                                  0045
```

```
                                 0045  ;+
                                 0045  ;       GET_ADP
                                 0045  ;
                                 0045  ; CALLING SEQUENCE:
                                 0045  ;
                                 0045  ;       $CMEXEC GET_ADP
                                 0045  ;
                                 0045  ; INPUT:
                                 0045  ;
                                 0045  ;       R8: nexus number to search for
                                 0045  ;
                                 0045  ; OUTPUT:
                                 0045  ;
                                 0045  ;       R6 = address of ADP CSR for that nexus
                                 0045  ;
                                 0045  ; RETURN STATUS:
                                 0045  ;
                                 0045  ;       SS$_NORMAL     - R6 is set appropriately
                                 0045  ;       SYSG$_NEXNOTUBA - specified nexus is not a UBA
                                 0045  ;
                                 0045  ;-
                                 0045
                             00000B79          .PSECT  PAGED_CODE      rd,nowrt,exe,long
                                 0B79
                           0000  0B79  GET_ADP:        .word 0                 ; Entry mask
                                 0B7B
            50     01      D0    0B7B          MOVL    #SS$_NORMAL,R0          ; Assume success
  52   00000000'GF         D0    0B7E          MOVL    G^IOC$GL_ADPLIST,R2     ; Get ADP list header
                                 0B85
        0C A2      58      B1    0B85  10$:    CMPW    R8,ADP$W_TR(R2)         ; Loop looking for nexus
                   08      13    0B89          BEQL    15$                     ; Branch if found match
        52      04 A2      D0    0B8B          MOVL    ADP$L_LINK(R2),R2       ; Get next ADP
                   F4      12    0B8F          BNEQ    10$                     ; Branch if not end of list
                                 0B91
                   09      11    0B91          BRB     20$                     ; End of list, no nexus found
                                 0B93
            56     62      D0    0B93  15$:    MOVL    ADP$L_CSR(R2),R6        ; Get ADP CSR
                   01      B1    0B96          CMPW    #AT$_UBA,-
                0E A2            0B98                  ADP$W_ADPTYPE(R2)       ; Make sure it's a UBA
                   07      13    0B9A          BEQL    30$                     ; Branch if OK
  50   007C80E2 8F         D0    0B9C  20$:    MOVL    #SYSG$_NEXNOTUBA,R0     ; Set error
                           04    0BA3  30$:    RET                             ; Return
                                 0BA4
                             0BA4  ;+
                             0BA4  ; SIGNAL_R0
                             CBA4  ;
                             0BA4  ; Call LIB$SIGNAL with R0 as the error if low bit is clear in R0.
                             0BA4  ;
                             0BA4  ;-
                             0BA4
                             0BA4  SIGNAL_R0:
                             0BA4
               0E 50   E8    0BA4          BLBS    R0,10$                  ; Branch if no error
                  50   DD    0BA7          PUSHL   R0                      ; Save R0
                  50   DD    0BA9          PUSHL   R0                      ; Set up for signal
     00000000'GF  01   FB    0BAB          CALLS   #1,G^LIB$SIGNAL         ; Signal message
               50 8ED0       0BB2          POPL    R0                      ; Restore R0
                       05    0BB5  10$:    RSB
                             0BB6
                             0BB6          .END
```

```
$$$CNT                          = 00000003              COMBO_VECTOR                      00000322 R     02
$$$FLG                          = FFFFFFFF              COMBO_VECTOR_OFFSET               00000326 R     02
$$$KEY                          = FFFFFFFF              CONF_PR                           000001AF R     02
$$$KFG                          = FFFFFFFF              CONNECT_OTHER                     000004D1 R     02
$$$MOD                          = 00000000              CONNECT_UBA                       0000040E R     02
$$.TMP1                         = 00000001              CONNECT_UBA2                      0000045E R     02
$$.TMP2                         = 000000EF              CRB$C_LENGTH                    = 00000048
$$KEYTAB                        = 00000000 R     04     CRB$L_INTD                      = 00000024
$$T2                            = 00000006              CRB$L_INTD2                     = 00000048
$EQV_DESC$                      = 00000050 R     09     CRB$W_SIZE                      = 00000008
$NAME1$                         = 00000080 R     07     CSR                               00000316 R     02
$NAME2$                         = 0000007E R     08     DDB$L_LINK                      = 00000000
AB_EQV_TABLE                      00000000 RG    06     DDB$L_UCB                       = 00000004
ACF$AB_UBATABLE                   ********    X  0A     DDB$T_DRVNAME                   = 00000024
ACF$INC_CHAR                      ********    X  0A     DDB$T_NAME                      = 00000014
ACF_NAME                          000000F9 R     02     DDB1                              00000705 R     0A
ADDRESS                         = 0000000C              DDBLOOP                           000006FD R     0A
ADDRESS_CALC                      00000252 R     0A     DEVICE                            0000000E R     03
ADP$L_CSR                       = 00000000              DEVICES                           00000089 R     02
ADP$L_LINK                      = 00000004              DEV_FOUND                         00000342 R     02
ADP$L_VECTOR                    = 00000010              DEV_LINE                          00000000 R     02
ADP$W_ADPTYPE                   = 0000000E              DONE                              00000A90 R     0A
ADP$W_TR                        = 0000000C              DR11B                             0000007A R     02
ADP_TYPE                          0000032E R     02     DRIVER                            0000033E R     02
AL_ADP_TEXT                       00000534 R     02     EXE$GL_NUMNEXUS                   ********    X  0A
AT$_CI                          = 00000004              EXE$TEST_CSR                      ********    X  0B
AT$_DR                          = 00000002              EXEC                              000006EC RG    0A
AT$_MBA                         = 00000000              EXIT                              0000051E R     0A
AT$_MPM                         = 00000003              FAO_D_CTRSTR                      00000253 R     02
AT$_NULL                        = 00000005              FAO_D_OUTBUF                      000001CB R     02
AT$_UBA                         = 00000001              FAO_Q_ONEUBA                      0000028D R     02
AVECT1                            0000030A R     02     FAO_W_OUTLEN                      000002C1 R     02
AVECT2                            0000030E R     02     FIRST                           = 00000010
BOO$CONFIGURE                     00000134 RG    0A     FL                                00000107 R     02
BOO$GL_CMDOPT                     ********    X  0A     FL_FL                             00000498 R     0A
BOO$GL_TR                         000001BF RG    02     FX                                00000105 R     02
BOO$NO_RESET                      00000000 RG    0A     FX_FL                             00000421 R     0A
BOO$OPEN_INPUT_2                  ********    X  0A     FX_FX                             000003DC R     0A
BOO$OPEN_OUTPUT_2                 ********    X  0A     GET_ADP                           00000B79 R     0A
BOO$SET_TR                        0000000B RG    0A     IDB$B_COMBO_CSR_OFFSET          = 0000000F
BOO$SHOCONFIG                     0000063C RG    0A     IDB$B_COMBO_VECTOR_OFFSET       = 00000010
BOO$SHOW_UNIBUS                   00000A94 RG    0A     IDB$B_VECTOR                    = 0000000B
BOOCMD$V_INPUT                  = 0000000E              IDB$L_ADP                       = 00000014
BOOCMD$V_NORESET                = 00000001              IDB$L_CSR                       = 00000000
BOOCMD$V_OUTPUT                 = 0000000D              IOC$AUTORESET                     ********    X  0A
BOOCMD$V_SAVE                   = 00000002              IOC$GL_ADPLIST                    ********    X  0A
BUF                               000001D3 R     02     IOC$GL_DEVLIST                    ********    X  0A
BUFFER_SIZE                     = 00000080              LENGTH                          = 00000008
B_CNUMVEC                         0000011C R     02     LIB$GET_INPUT                     ********    X  0A
CHECK_CSR                         00000000 R     0B     LIB$SIGNAL                        ********    X  0A
CNF$FIND_DEVICE                   00000081 RG    0A     LIB$TPARSE                        ********    X  0A
CNF$KEYTBL                        00000000 RG    04     LIB$_SYNTAXERR                    ********    X  0A
CNF$PREV_UNIBUS                   00000016 RG    0A     LOOKUP                            000000DF RG    0A
CNF$SET_VALUE                     0000003A RG    0A     LOOP                              00000386 R     0A
CNF$STATE                         00000000 RG    03     LPA11                             00000074 R     02
COMBO_CSR                         0000031A R     02     L_DEVNAME                         000000ED R     02
COMBO_CSR_OFFSET                  0000031E R     02     L_DRVNAME                         000000F1 R     02
```

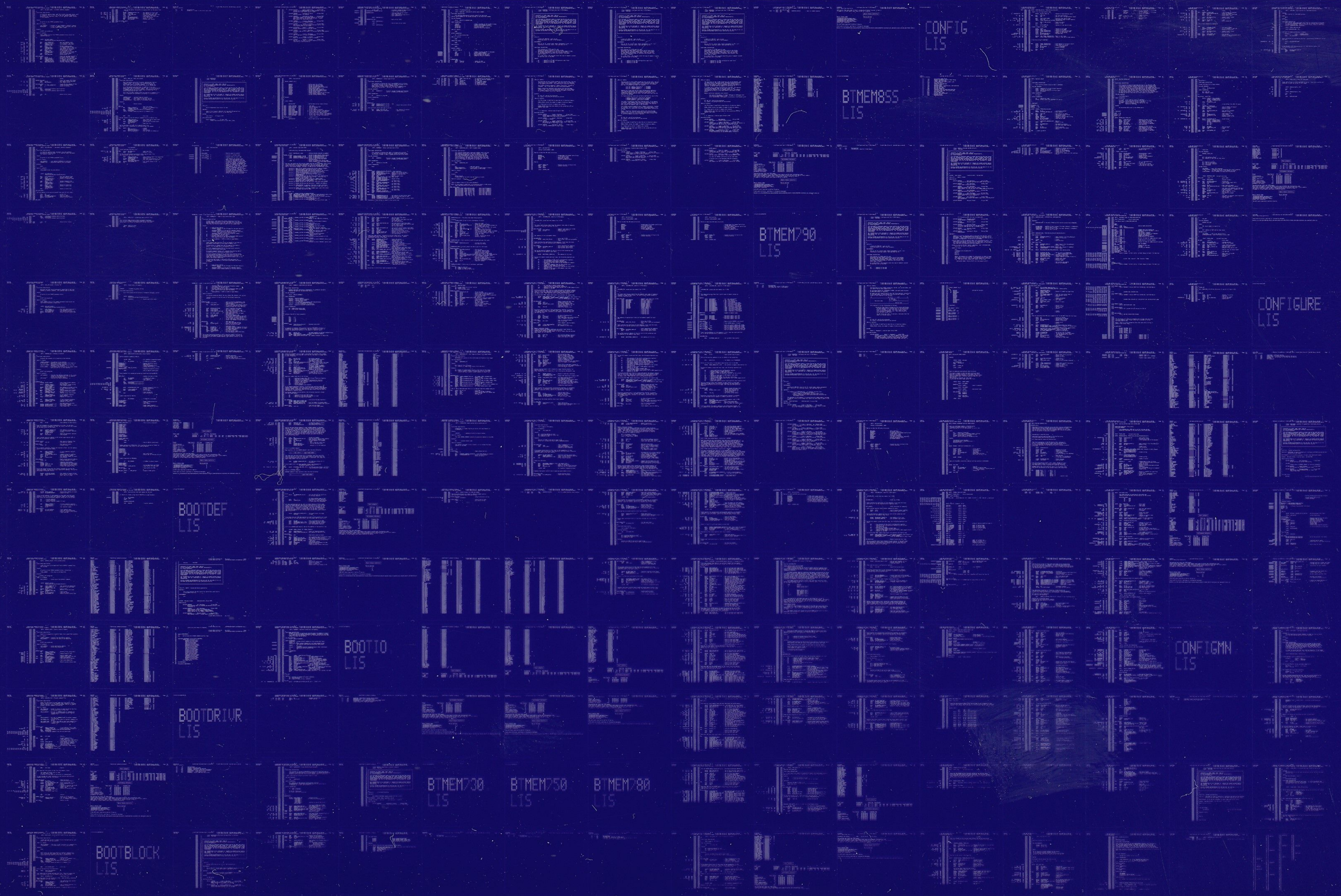
```
L_MAXADP                         00000550 R      02     SYSG$_DEVNOTKNWN            = 007C9008
L_ROUTINE                        000000F5 R      02     SYSG$_ENDUBA                = 007C9040
MMG$GL_SPTBASE                   ********   X    0A     SYSG$_EQV_NOTICE            = 007CA003
NAME                             000002E7 R      02     SYSG$_INPUT_ERR             = 007C80A2
NAME_L                           000002E7 R      02     SYSG$_NEXNOTUBA             = 007C80E2
NAME_S                           000002EF R      02     SYSG$_NODEVADAP             = 007C9020
NO                               0000010D R      02     SYSG$_OUT_RANGE             = 007C9000
NULL                             000001C3 R      02     SYSG$_OVERFLOW              = 007C8092
NUM                              00000101 R      02     SYSG$_RSV_ERR               = 007C9018
NUMBER                           00000016 R      03     SYSG$_SKIPPED               = 007CA02B
NUMBER2                          0000002A R      03     SYSG$_SYNTAX                = 007C809A
NVECT                            00000302 R      02     SYSG$_TOO_MNY               = 007C80AA
OCC1                           = 0000000C               SYSG$_TWICE                 = 007CA00B
OCC2                           = 00000004               TOO_MANY                      00000509 R      0A
OCCURANCE                      = 00000004               TPA$K_COUNT0                = 00000008
OFFSET                           000000FD R      02     TPA$K_LENGTH0               = 00000024
ONE_UNIBUS                       00000AF2 R      0A     TPA$L_COUNT                 = 00000000
OTHER_BLOCK                      00000343 R      02     TPA$L_NUMBER                = 0000001C
OUTPUT_DESC                      0000011F R      02     TPA$L_OPTIONS               = 00000004
OVECT1                           00000332 R      02     TPA$L_PARAM                 = 00000020
OVECT2                           00000336 R      02     TPA$L_STRINGCNT             = 00000008
PARAM_BLK                        000002C3 R      02     TPA$L_STRINGPTR             = 0000000C
PR$_IPL                        = 00000012               TPA$L_TOKENCNT              = 00000010
PRINT                            00000884 R      0A     TPA$L_TOKENPTR              = 00000014
PUT_LINE                         00000549 RG     0A     TPA$M_ABBREV                = 00000002
RAB$W_RSZ                        ********   X    0A     TPA$_ALPHA                  = 000001EE
READ_PARSE_INPUT                 00000189 R      0A     TPA$_ANY                    = 000001ED
REARNG_DEV                       000005D0 RG     0A     TPA$_BLANK                  = 000001F2
RIO$AB_INBUFFER                  ********   X    0A     TPA$_DECIMAL                = 000001F3
RIO$AB_OUTBUF                    ********   X    0A     TPA$_DIGIT                  = 000001EF
RIO$GW_OUTLEN                    ********   X    0A     TPA$_EOS                    = 000001F7
RIO$OUTPUT_LINE                  ********   X    0A     TPA$_EXIT                   = FFFFFFFF
RIO_INFAB2                       ********   X    0A     TPA$_FAIL                   = FFFFFFFE
RIO_INRAB2                       ********   X    0A     TPA$_FILESPEC               = 000001EA
RIO_OUTFAB2                      ********   X    0A     TPA$_HEX                    = 000001F5
RL11                             0000005A R      02     TPA$_IDENT                  = 000001EC
RL211                            0000005F R      02     TPA$_KEYWORD                = 00000100
RMS$_EOF                         ********   X    0A     TPA$_LAMBDA                 = 000001F6
RSV                              00000057 R      02     TPA$_MAXKEY                 = 000000DC
RX211                            0000006A R      02     TPA$_OCTAL                  = 000001F4
SAVE_HEADER                      00000515 R      02     TPA$_STRING                 = 000001F0
SAVE_OUTPUT                      000009A4 R      0A     TPA$_SUBXPR                 = 000001F8
SECOND                         = 00000008               TPA$_SYMBOL                 = 000001F1
SHOCON_HEADER                    000004EA R      02     TPA$_UIC                    = 000001EB
SHOW_OTHER                       000003CC R      02     TR                            0000032A R      02
SHOW_UBA                         00000363 R      02     TS11                          00000065 R      02
SIGNAL_RO                        00000BA4 R      0A     TU81                          00000084 R      02
SPACE                          = 00000020               T_NVECT                       00000306 R      02
SS$_NORMAL                     = 00000001               UBA_IOBASE                  = 00001000
STR$UPCASE                       ********   X    0A     UBA_M_FLOATCSR              = 00000002
SUP                              00000110 R      02     UBA_M_FLOATVEC              = 00000004
SYS$CLOSE                        ********  GX    0A     UBA_M_SUPPORT               = 00000001
SYS$CMEXEC                       ********  GX    0A     UBA_V_FLOATCSR              = 00000001
SYS$CMKRNL                       ********  GX    0A     UBA_V_FLOATVEC              = 00000002
SYS$FAO                          ********   X    0A     UBA_V_SUPPORT               = 00000000
SYS$GET                          ********  GX    0A     UBT$B_FLAGS                   ********   X    0A
SYSG$_ABORT                    = 007C8082               UBT$B_NUMVEC                  ********   X    0A
```

```
UBT$L_DEVNAME                    ********   X    0A
UBT$L_DRVNAME                    ********   X    0A
UBT$L_RTNNAME                    ********   X    0A
UBT$W_REMAINDER                  ********   X    0A
UCB$L_CRB                      = 00000024
UCB$L_LINK                     = 00000030
UCB$W_UNIT                     = 00000054
UCB1                             00000746 R      0A
UCB_SAVE                         0000033A R      02
UDA                              00000070 R      02
UNA                              00000080 R      02
UNIT                             000002FE R      02
UPCASE_DST                       00000127 R      02
UPCASE_SRC                       0000011F R      02
VCSR                             00000312 R      02
VEC$C_LENGTH                   = 00000024
VEC$L_IDB                      = 00000008
VECT1                            00000776 R      0A
W_CSRBASE                        00000114 R      02
W_VECBASE                        00000118 R      02
W_VECMOD                         0000011D R      02
YES                              00000109 R      02

                                           +----------------+
                                           ! 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
PAGED_DATA                      00000554  ( 1364.)  02 (  2.)  NOPIC    USR    CON    REL    LCL NOSHR NOEXE   RD     WRT NOVEC QUAD
_LIB$STATE$                     0000003A  (    58.)  03 (  3.)    PIC    USR    CON    REL    LCL   SHR   EXE   RD   NOWRT NOVEC BYTE
_LIB$KEY0$                      00000000  (     0.)  04 (  4.)    PIC    USR    CON    REL    LCL   SHR   EXE   RD   NOWRT NOVEC WORD
_LIB$KEY1$                      00000000  (     0.)  05 (  5.)    PIC    USR    CON    REL    LCL   SHR   EXE   RD   NOWRT NOVEC WORD
EQV_DATA                        00000030  (    48.)  06 (  6.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
ACF_NAMES                       0000008D  (   141.)  07 (  7.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
EQV_NAMES                       0000008B  (   139.)  08 (  8.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
EQV_DESC                        00000058  (    88.)  09 (  9.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
PAGED_CODE                      00000BB6  ( 2998.)  0A ( 10.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC LONG
NONPAGED_CODE                   00000045  (    69.)  0B ( 11.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD   NOWRT NOVEC LONG

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

Phase                        Page faults    CPU Time       Elapsed Time
-----                        -----------    --------       ------------
Initialization                        30    00:00:00.07    00:00:00.31
Command processing                   110    00:00:00.64    00:00:01.85
Pass 1                               546    00:00:22.83    00:00:42.65
Symbol table sort                      0    00:00:02.59    00:00:03.03
Pass 2                               305    00:00:06.12    00:00:14.74
Symbol table output                   30    00:00:00.22    00:00:00.22
Psect synopsis output                  4    00:00:00.06    00:00:00.14
Cross-reference output                 0    00:00:00.00    00:00:00.00
Assembler run totals                1027    00:00:32.53    00:01:02.94
```

```
The working set limit was 1950 pages.
135235 bytes (265 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1639 non-local and 106 local symbols.
1731 source lines were read in Pass 1, producing 78 object records in Pass 2.
54 pages of virtual memory were used to define 46 macros.

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

Macro library name                                   Macros defined
------------------                                   --------------
_$255$DUA28:[BOOTS.OBJ]BOOTS.MLB;1                                1
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                                   10
_$255$DUA28:[SYSLIB]STARLET.MLB;2                                21
TOTALS (all libraries)                                           32

1957 GETS were required to define 32 macros.

There were no errors, warnings or information messages.

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

0037 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



CONFIG
LIS

BTMEM8SS
LIS

BTMEM790
LIS

CONFIGURE
LIS

BOOTDEF
LIS

BOOTIO
LIS

CONFIGMN
LIS

BOOTDRIVR
LIS

BTMEM730
LIS

BTMEM750
LIS

BTMEM780
LIS

BOOTBLOCK
LIS