IDENTIFICATION

Product code: ENKAZ VERSION 1.3

Product name: CRD HARDCORE INSTRUCTION and SIZING TESTS

Product date: 23-MAY-1983

Maintainer: BASE SYSTEMS DIAGNOSTIC ENGINEERING

Table of Contents

1 PROGRAM ABSTRACT

This Level 4 diagnostic is run during VAX-11/730/725 Customer Runable Diagnostics AUTO and MENU mode. Its primary functions are:

1) Providing a small suoset of the VAX Macro-Hardcore Instruction Set prior to booting the Diagnostic Supervisor.

2) Sizing base VAX-11/730/725 disk system load devices, (via IDC, UDA50 or LESI controllers), using an algorithm which is defined in ENSAB.DOC for AUTO and MENU CRD. Sizing is done to pass the corrcct parameters through VMB.EXE to boot the Diagnostic Supervisor on the correct load device.

3) Verification of LESI and UDA50 CRD supported load disk options by testing if the VAX-11/730/725 CRD suppported load device (RA80, RA81, RA60, or RC25) is spinning and reading "CRDPACK" label for the RA60 and RC25.

This diagnostic was designed to be run on the VAX-11/730/725 specifically and it is not intended to be APT compatible. Refer to ENSAB.DOC the CRD AUTO and MENU mode documentation for further information on this diagnostic and complete documentation on VAX-11/730/725 CRD AUTO test.

2 MAINTENANCE HISTORY

DATE	VERSION	DESCRIPTION
23-MAY-1983	1.0	Initial release.
25-JUL-1983	1.1	Added LCN VAX-11/725 syntax.
16-JAN-1984	1.2	Added code for detection of IDC and detection of drive 1 not present in which we boot from drive 0 European system configuration.
02-FEB-1984	1.2	Added code for printing on the VR1C0 terminal in simulated VT52 mode. LCN RL25 system configuration.
14-Jun-1984	1.3	Code for booting from drive 0 when no RL02 is present was fixed. The 730/725 processor register definitions were added The RC25 based systems can now go straight into menu mode.

Object Module Synopsis

Module Name	Ident	Bytes	File	Creation Date	Creator
ENKAZ01	01-30	5080	[BALL.ENKAZ.ENKAZ_1]ENKAZ01A.OBJ;5	14-JUN-1984 16:01	VAX-11 Macro V03-01
ENKAZ02	01-30	2265	[BALL.ENKAZ.ENKAZ_1]ENKAZ02A.OBJ;12	14-JUN-1984 16:04	VAX-11 Macro V03-01

Image Section Synopsis

Cluster	Type	Pages	Base Addr	Disk	VBN	PFC	Protection and Paging	Global Sec. Name	Match	Majorid	Minorid
DEFAULT_CLUSTER	0	15	00000000		1	0	READ ONLY				

Key for special characters above:

R	- Relocatable
P	- Protected

Program Section Synopsis

Psect Name	Module Name	Base	End	Length	Align	Attributes
A_CONIO		00000000	0000125F	00001260 (4704.)	PAGE 9	PIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ENKAZ01	00000000	0000125F	00001260 (4704.)	PAGE 9	
A_HARDCORE		00001260	000013D7	00000178 (376.)	BYTE 0	PIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ENKAZ01	00001260	000013D7	00000178 (376.)	BYTE 0	
SIZING		00001400	00001CD8	000008D9 (2265.)	PAGE 9	NOPIC,USR,CON,REL,LCL,NOSHR, EXE, RD, WRT,NOVEC
	ENKAZ02	00001400	00001CD8	000008D9 (2265.)	PAGE 9	

Symbol Cross Reference

Symbol	Value	Defined By	Referenced By ...
BAD_DISK	00001688-R	ENKAZ02	ENKAZ01
BOO$READPROMPT	00000E00-R	ENKAZ01	
C_D_RDY	00000081	ENKAZ02	
DRIVE0	000017B5-R	ENKAZ02	
DS$GB_UDA50_INIT	00001480-R	ENKAZ02	ENKAZ01
DS1	00000100	ENKAZ02	
ERROR_ROUTINE	00000F29-R	ENKAZ01	ENKAZ02
HMBLK_ERR	00001687-R	ENKAZ02	ENKAZ01
IDC_CSR	00F26200	ENKAZ01	ENKAZ02
IDC_HLD	0000168A-R	ENKAZ02	
IDC_PARAM_0	00001784-R	ENKAZ02	
IDC_PARAM_1	0000175C-R	ENKAZ02	
IDC_TST	0000168E-R	ENKAZ02	
INIT_DRV0	00001847-R	ENKAZ02	ENKAZ01
INTERRUPT_STACK	00000A00-R	ENKAZ01	
KERNEL_STACK	00000800-R	ENKAZ01	
LESI_UDA_PARAM	00001736-R	ENKAZ02	ENKAZ01
MCHK_IDC	000011F4-R	ENKAZ01	ENKAZ02
MCHK_LES_UDA	00001208-R	ENKAZ01	ENKAZ02
NOCNTLR_FND	00000207-R	ENKAZ01	ENKAZ02
NODRIVE_0_1	00001686-R	ENKAZ02	ENKAZ01
NO_LOAD	00000F6A-R	ENKAZ01	
RA60_TEST	00000FE0-R	ENKAZ01	ENKAZ02
RA8081_TEST	0000109E-R	ENKAZ01	ENKAZ02
RA81_PRES	00001684-R	ENKAZ02	ENKAZ01
RC25_PRES	00001685-R	ENKAZ02	
RC25_TEST	00001121-R	ENKAZ01	ENKAZ02
RPB_BUFF	00000C00-R	ENKAZ01	ENKAZ02
SCB	00000600-R	ENKAZ01	
SCB0	00000600-R	ENKAZ01	
SCB20	00000620-R	ENKAZ01	
SCB24	00000624-R	ENKAZ01	
SCB4	00000604-R	ENKAZ01	ENKAZ02
SCB60	00000660-R	ENKAZ01	
SCB64	00000664-R	ENKAZ01	
SCB8	00000608-R	ENKAZ01	
SPIN_DWN	00001689-R	ENKAZ02	ENKAZ01
TEST_6	000016E3-R	ENKAZ02	ENKAZ01
TEST_7	000017AC-R	ENKAZ02	
TEST_8	00001C00-R	ENKAZ02	ENKAZ01
TOP_OF_ISTACK	00000BFC-R	ENKAZ01	
TOP_OF_KSTACK	000009FC-R	ENKAZ01	
UDA_LESI	0000171F-R	ENKAZ02	ENKAZ01
UDA_LESI_CSR	40FFF468	ENKAZ01	ENKAZ02
UDA_RESPONSE	00001650-R	ENKAZ02	
UD_DRIVER1	00001692-R	ENKAZ02	ENKAZ01
UD_INIT	00001400-R	ENKAZ02	
UNEX_E_I	000011C8-R	ENKAZ01	ENKAZ02
WHAT_DRIVE	00001680-R	ENKAZ02	ENKAZ01

Symbol	Value	Defined By	Referenced By ...
WHAT_MODE	00000203-R	ENKAZ01	ENKAZ02

Symbols By Value

Value	Symbols...	
00000081	C_D_RDY	
00000100	DS1	
00000203	R-WHAT_MODE	
00000207	R-NOCNTLR_FND	
000006C0	R-SCB	R-SCB0
00000604	R-SCB4	
00000608	R-SCB8	
00000620	R-SCB20	
00000624	R-SCB24	
00000660	R-SCB60	
00000664	R-SCB64	
00000800	R-KERNEL_STACK	
000009FC	R-TOP_OF_KSTACK	
00000A00	R-INTERRUPT_STACK	
00000BFC	R-TOP_OF_ISTACK	
00000C00	R-RPB_BUFF	
00000E00	R-BOO$READPROMPT	
00000F29	R-ERROR_ROUTINE	
00000F6A	R-NO_LOAD	
00000FE0	R-RA60_TEST	
0000109E	R-RA8081_TEST	
00001121	R-RC25_TEST	
000011C8	R-UNEX_E_I	
000011F4	R-MCHK_IDC	
00001208	R-MCHK_LES_UDA	
00001400	R-UD_INIT	
000014B0	R-DS$GB_UDA50_INIT	
00001650	R-UDA_RESPONSE	
00001680	R-WHAT_DRIVE	
00001684	R-RA81_PRES	
00001685	R-RC25_PRES	
00001686	R-NODRIVE_0_1	
00001687	R-HMBLK_ERR	
00001688	R-BAD_DISK	
00001689	R-SPIN_DWN	
0000168A	R-IDC_ALD	
0000168E	R-IDC_TST	
00001692	R-UD_DRIVER1	
000016E3	R-TEST_6	
0000171F	R-UDA_LESI	
00001736	R-LESI_UDA_PARAM	
0000175C	R-IDC_PARAM_1	
00001784	R-IDC_PARAM_0	
000017AC	R-TEST_7	
00001785	R-DRIVE0	
00001847	R-INIT_DRV0	
00001C00	R-TEST_8	
00F26200	IDC_CSR	
40FFF468	UDA_LESI_CSR	

Value Symbols...

Key for special characters above:

*	- Undefined
U	- Universal
R	- Relocatable
X	- External

Image Synopsis

```
Virtual memory allocated:                00000000 00001DFF 00001E00 (7680. bytes, 15. pages)
Stack size:                                     0. pages
Image binary virtual block limits:              1.      15. (   15. blocks)
Image name and identification:           ENKAZ 01-30
Number of files:                                2.
Number of modules:                              2.
Number of program sections:                     5.
Number of global symbols:                      50.
Number of cross references:                    76.
Number of image sections:                       1.
Image type:                              SYSTEM.
Map format:                              FULL WITH CROSS REFERENCE in file DRB0:[BALL.ENKAZ.ENKAZ_1]ENKAZ.MAP,13
Estimated map length:                    26. blocks
```

Link Run Statistics

Performance Indicators	Page Faults	CPU Time	Elapsed Time
Command processing	102	00:00:00.29	00:00:01.87
Pass 1:	35	00:00:00.17	00:00:00.70
Allocation/Relocation:	10	00:00:00.14	00:00:01.17
Pass 2:	13	00:00:00.22	00:00:01.43
Map data after object module synopsis:	19	00:00:00.40	00:00:03.10
Symbol table output:	0	00:00:00.11	00:00:01.07
Total run values:	179	00:00:01.33	00:00:09.34

```
...g a working set limited to 500 pages and 33 pages of data storage (excluding image)

Total number object records read (both passes):   90
    of which 0 were in libraries and 4 were DEBUG data records containing 145 bytes

Number of modules extracted explicitly              = 0
    with 0 extracted to resolve undefined symbols

0 library searches were for symbols not in the library searched

A total of 5 global symbol table records was written

LINK/SYMBOL=ENKAZ/NODEBUG/MAP=ENKAZ/FULL/CROSS/SYSTEM=%X0/EXE=ENKAZ ENKAZ01A,ENKAZ02A
```

Table of contents

```
                        0000     1          .TITLE  ENKAZ01 CRD HARDCORE INSTRUCTION TESTS
                        0000     2          .IDENT  /01-30/
                        0000     3
                        0000     4 ;
                        0000     5 ; Copyright (c) 1982
                        0000     6 ; Digital Equipment Corporation, Maynard, Massachusetts 01754
                        0000     7 ;
                        0000     8 ; This software is furnished under a license for use only on a single computer
                        0000     9 ; system and may be copied only with the inclusion of the above copyright
                        0000    10 ; notice.  This software, or any other copies thereof, may not be provided or
                        0000    11 ; otherwise made available to any other person except for use on such system
                        0000    12 ; and to one who agrees to these license terms.  Title to and ownership of the
                        0000    13 ; software shall at all times remain in DEC.
                        0000    14 ;
                        0000    15 ; The information in this software is subject to change without notice and
                        0000    16 ; should not be construed as a commitment by Digital Equipment Corporation.
                        0000    17 ;
                        0000    18 ; DEC assumes no responsibility for the use or reliability of its software on
                        0000    19 ; equipment which is not supplied by DEC.
                        0000    20 ;
                        0000    21
                        0000    22 ;++
                        0000    23 ; Facility:
                        0000    24 ;
                        0000    25 ;       Vax Diagnostic Library
                        0000    26 ;
                        0000    27 ; Abstract:
                        0000    28 ;
                        0000    29 ;       This diagnostic provides level 4 coverage while executing Customer
                        0000    30 ;       Runable Diagnostics. Its primary functions are:
                        0000    31 ;
                        0000    32 ;       1) Providing a small subset of the VAX Macro-Hardcore Instruction
                        0000    33 ;          Set prior to booting the Diagnostic Supervisor.
                        0000    34 ;
                        0000    35 ;       2) Sizing base VAX-11/730 disk system load devices via IDC, UDA50 or
                        0000    36 ;          LESI using an algorithm which is defined in ENSAB.DOC for AUTO and
                        0000    37 ;          MENU CRD. Sizing is done to pass the correct parameters through
                        0000    38 ;          VMB.EXE to boot the Diagnostic Supervisor on the correct load
                        0000    39 ;          device. This is done utilizing PUBTDRIVER module.
                        0000    40 ;
                        0000    41 ;       3) Verification of KLESI and UDA50 CRD supported load disk options by
                        0000    42 ;          testing if the  VAX-11/730 CRD suppported load device (RA80, RA81,
                        0000    43 ;          RA60, or RC25) is spinning and reading "CRDPACK" label for the RA60
                        0000    44 ;          and RC25.
                        0000    45 ;
                        0000    46 ; Environment:
                        0000    47 ;
                        0000    48 ;       Stand-alone
                        0000    49 ;
                        0000    50 ; Author:
                        0000    51 ;
                        0000    52 ;       Peter Green     23-May-83       Version 1.0
                        0000    53 ;
                        0000    54 ;       Peter Green     5-JUL-83        Version 1.1
                        0000    55 ;       Added LCN syntax.
                        0000    56 ;
                        0000    57 ;
```

```
                        0000    58 ;--
```

```
                        0000     60
                        0000     61
                        0000     62 ;
                        0000     63 ; Revision
                        0000     64 ;
                00000001 0000    65 PRIM_REV          =       1
                00000003 0000    66 SEC_REV           =       3
                        0000     67 ;
                        0000     68 ;
                        0000     69
```

```
                        71          .SBTTL  DECLARATIONS
                        72          .PSECT  A_CONIO,PAGE,PIC                ;
                        73
                        74          .LIST MEB
                        75
                        76 ;
                        77 ; Macro's
                        78 ;
                        79
                        80
                        81 .MACRO   PASS_MESSAGE
                        82          .ASCIC  /    PASSED/<13><10><13><10>
                        83 .ENDM    PASS_MESSAGE
                        84
                        85 .MACRO   FAIL_MESSAGE
                        86          .ASCIC  /    *FAILED*/<13><10><13><10><13><10>
                        87 .ENDM    FAIL_MESSAGE
                        88
                        89 .MACRO   ABORT_MESSAGE
                        90          .ASCIC  /TEST ABORTED - /
                        91 .ENDM    ABORT_MESSAGE
                        92
                        93 .MACRO   CPU_MESSAGE
                        94          .ASCIC  /CPU BAD - /
                        95 .ENDM    CPU_MESSAGE
                        96
                        97 .MACRO   FIELD_MESSAGE
                        98          .ASCIC  /CALL FIELD SERVICE/
                        99 .ENDM    FIELD_MESSAGE
                       100
                       101 .MACRO   NOCNTLR_MESSAGE
                       102          .ASCIC  /NO SUPPORTED DISK CONTROLLER FOUND/<13><10>
                       103 .ENDM    NOCNTLR_MESSAGE
                       104
                       105 .MACRO   NODRIVE01_MESSAGE
                       106          .ASCIC  /UNSUPPORTED DISK CONFIGURATION FOUND - DRIVE 0 AND 1/<13><10>
                       107 .ENDM    NODRIVE01_MESSAGE
                       108
                       109 .MACRO   LOADMCK_MESSAGE
                       110          .ASCIC  /UNEXPECTED MACHINE CHECK WHILE VERIFYING LOAD PATH/<13><10>
                       111 .ENDM    LOADMCK_MESSAGE
                       112
                       113 .MACRO   DISKERR_MESSAGE
                       114          .ASCIC  /DRIVE BAD - /
                       115 .ENDM    DISKERR_MESSAGE
                       116
                       117 .MACRO   END_MESSAGE
                       118          .ASCIC  <13><10>?**** END OF VAX-11/730,725 ?
                       119 .ENDM    END_MESSAGE
                       120
                       121 .MACRO   TEND_MESSAGE
                       122          .ASCIC  ?TEST ****?<13><10><13><10>
                       123 .ENDM    TEND_MESSAGE
                       124
                       125 .MACRO   RET_MESSAGE
                       126          .ASCIC  ?RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR '>>>' PROMPT?-
                       127                  <13><10>
```

```
0000    128 .ENDM   RET_MESSAGE
0000    129
0000    130 .MACRO  PROPER_MESSAGE
0000    131         .ASCIC  <13><10><13><10>/**       PROPER SETUP REQUIRED          **/
0000    132 .ENDM   PROPER_MESSAGE
0000    133
0000    134 .MACRO  NSPN6025_MESSAGE
0000    135         .ASCIC  <13><10>/**  CHECK FOR DIAGNOSTIC PACK CRDPACK  **/-
0000    136                 <13><10>/**    INSERTED AND SPINNING IN /
0000    137 .ENDM   NSPN6025_MESSAGE
0000    138
0000    139 .MACRO  DRVERR_MESSAGE
0000    140         .ASCIC  /**  IF SETUP IS CORRECT, SUSPECT DRIVE  **/<13><10>-
0000    141                 /**      PROBLEM - /
0000    142 .ENDM   DRVERR_MESSAGE
0000    143
0000    144 .MACRO  SPNEND_MESSAGE
0000    145         .ASCIC  /    **/<13><10>
0000    146 .ENDM   SPNEND_MESSAGE
0000    147
0000    148 .MACRO  NSPN8081_MESSAGE
0000    149         .ASCIC  <13><10>/**     CHECK THAT DUA0: IS SPINNING     **/<13><10>
0000    150 .ENDM   NSPN8081_MESSAGE
0000    151
0000    152 .MACRO  AUTO_MESSAGE
0000    153         .ASCIC  /AUTO /
0000    154 .ENDM   AUTO_MESSAGE
0000    155
0000    156 .MACRO  MENU_MESSAGE
0000    157         .ASCIC  /MENU /
0000    158 .ENDM   MENU_MESSAGE
0000    159
0000    160 .MACRO  TEST_MESSAGE
0000    161         .ASCIC  /TESTING STARTED - RUN TIME = 0:30 MINUTES/
0000    162 .ENDM   TEST_MESSAGE
0000    163
0000    164 .MACRO  PART_MESSAGE
0000    165         .ASCIC  /PART 1    /
0000    166 .ENDM   PART_MESSAGE
0000    167
0000    168 .MACRO  RC25_MESSAGE
0000    169         .ASCIC  /RC25 /
0000    170 .ENDM   RC25_MESSAGE
0000    171
0000    172 .MACRO  RA60_MESSAGE
0000    173         .ASCIC  /RA60 /
0000    174 .ENDM   RA60_MESSAGE
0000    175
0000    176 .MACRO  RA80_MESSAGE
0000    177         .ASCIC  /RA80 /
0000    178 .ENDM   RA80_MESSAGE
0000    179
0000    180 .MACRO  RA81_MESSAGE
0000    181         .ASCIC  /RA81 /
0000    182 .ENDM   RA81_MESSAGE
0000    183
0000    184 .MACRO  DAA0_MESSAGE
```

```
                        0000   185          .ASCIC  /DAA0 /
                        0000   186 .ENDM    DAA0_MESSAGE
                        0000   187
                        0000   188 .MACRO   DJA0_MESSAGE
                        0000   189          .ASCIC  /DJA0 /
                        0000   190 .ENDM    DJA0_MESSAGE
                        0000   191
                        0000   192 .MACRO   DJA1_MESSAGE
                        0000   193          .ASCIC  /DJA1 /
                        0000   194 .ENDM    DJA1_MESSAGE
                        0000   195
                        0000   196 .MACRO   DUA0_MESSAGE
                        0000   197          .ASCIC  /DUA0 /
                        0000   198 .ENDM    DUA0_MESSAGE
                        0000   199
                        0000   200 ;
                        0000   201 ; JMP to the starting address.
                        0000   202 ;
                00000200  0000   203          . = ^X200
                101D   31 0200   204          BRW     INIT1
                        0203   205
                        0203   206
                        0203   207 ;
                        0203   208 ; Equated symbols:
                        0203   209 ;
                        0203   210
                00000012  0203   211 IPL     =       18
                00000004  0203   212 MCKADD  =       4
                00000022  0203   213 TXCS    =       34
                00000023  0203   214 TXDB    =       35
                0000000D  0203   215 CR      =       13
                0000000A  0203   216 LF      =       10
                00F26200  0203   217 IDC_CSR          == ^XF26200
                40FFF468  0203   218 UDA_LESI_CSR     == ^X40FFF468
                00000084  0203   219 BOO$C_COMBUFSZ   =       132
                00000000  0203   220 WHAT_MODE::     .LONG 0
                00000000  0207   221 NOCNTLR_FND::    .LONG 0
                00000000  020B   222 LOAD_ERROR:      .LONG 0
                00000000  020F   223 CPU_BAD:         .LONG 0
                00000000  0213   224 PASS_COUNT:      .LONG 0
                00000000  0217   225 RA60_MESS_ST:    .LONG 0
                00000000  021B   226 RA8081_MESS_ST:  .LONG 0
                00000000  021F   227 RC25_MESS_ST:    .LONG 0
                          0223   228 PRINTPASS:       PASS_MESSAGE
0D 44 45 53 53 41 50 20 20 20 20 00'  0223                .ASCIC  /    PASSED/<13><10><13><10>
                   0A 0D 0A   022F
                      0E      0232
                          0232   229 PRINTFIELD:     FIELD_MESSAGE
20 44 4C 45 49 46 20 4C 4C 41 43 00'  0232                .ASCIC  /CALL FIELD SERVICE/
                45 43 49 56 52 45 53  023E
                         12   0245
                          0245   230 PRINTFAIL:      FAIL_MESSAGE
2A 44 45 4C 47 41 46 2A 20 20 20 00'  0245                .ASCIC  /   *FAILED*/<13><10><13><10><13><10>
                   0A 0D 0A 0D 0A 0D  0251
                         11   0245
                          0257   231 PRINTABORT:     ABORT_MESSAGE
45 54 52 4F 42 41 20 54 53 45 54 00'  0257                .ASCIC  /TEST ABORTED - /
```

```
                                                20 2D 20 44   0263
                                                      0F      0257
                                                              0267
20 44 4E 45 20 2A 2A 2A 2A 0A 0D 00'          0267
33 37 2F 31 31 2D 58 41 56 20 46 4F           0273
                  20 35 32 37 2C 30           027F
                                 1D           0267
                                              0285
0A 0D 2A 2A 2A 2A 20 54 53 45 54 00'          0285
                                 0A 0D        0291
                                    0D        0285
                                              0293
54 20 47 4E 49 4E 52 55 54 45 52 00'          0293
30 33 37 2F 31 31 2D 58 41 56 20 4F           029F
45 4C 4F 53 4E 4F 43 20 35 32 37 2C           02AB
27 20 52 4F 46 20 54 49 41 57 20 2C           02B7
0D 54 50 4D 4F 52 50 20 27 3E 3E 3E           02C3
                                 0A           02CF
                                 3C           0293
                                              02D0
55 54 45 53 20 46 49 20 20 2A 2A 00'          02D0
54 43 45 52 52 4F 43 20 53 49 20 50           02DC
52 44 20 54 43 45 50 53 55 53 20 2C           02E8
                  0D 2A 2A 20 20 45 56 49     02F4
4F 52 50 20 20 20 20 20 20 2A 2A 0A           02FC
                  20 2D 20 4D 45 4C 42        0308
                                 3E           02D0
                                              030F
2D 20 44 41 42 20 45 56 49 52 44 00'          030F
                                    20        031B
                                    0C        030F
                                              031C
20 44 45 54 43 45 50 58 45 4E 55 00'          031C
43 45 48 43 20 45 4E 49 48 43 41 4D           0328
49 52 45 56 20 45 4C 49 48 57 20 4B           0334
50 20 44 41 4F 4C 20 47 4E 49 59 46           0340
                        0A 0D 48 54 41        034C
                                 34           031C
                                              0351
45 54 52 4F 50 50 55 53 20 4F 4E 00'          0351
52 54 4E 4F 43 20 4B 53 49 44 20 44           035D
0D 44 4E 55 4F 46 20 52 45 4C 4C 4F           0369
                                    0A        0375
                                    24        0351
                                              0376
44 45 54 52 4F 50 50 55 53 4E 55 00'          0376
47 49 46 4E 4F 43 20 4B 53 49 44 20           0382
4E 55 4F 46 20 4E 4F 49 54 41 52 55           038E
20 30 20 45 56 49 52 44 20 2D 20 44           039A
                     0A 0D 31 20 44 4E 41     03A6
                                    36        0376
                                              03AD
20 20 20 20 20 2A 2A 0A 0D 0A 0D 00'          03AD
54 45 53 20 52 45 50 4F 52 50 20 20           03B9
09 44 45 52 49 55 51 45 52 20 50 55           03C5
      2A 2A 20 20 20 20 20 20 20 20           03D1
                                    2D        03AD
```

```
232 PRINTEND:       END_MESSAGE
                    .ASCIC  <13><10>?**** END OF VAX-11/730,725 ?

233 PRINTTEND:      TEND_MESSAGE
                    .ASCIC  ?TEST ****?<13><10><13><10>

234 PRINTRET:       RET_MESSAGE
                    .ASCIC  ?RETURNING TO VAX-11/730,725 CONSOLE, WAIT FOR '>>>' PROMPT?-

235 PRINTDRVERR:    DRVERR_MESSAGE
                    .ASCIC  /**  IF SETUP IS CORRECT, SUSPECT DRIVE  **/<13><10>-

                            /**       PROBLEM - /

236 PRINTDISKERR:   DISKERR_MESSAGE
                    .ASCIC  /DRIVE BAD - /

237 PRINTLOADMCK:   LOADMCK_MESSAGE
                    .ASCIC  /UNEXPECTED MACHINE CHECK WHILE VERIFYING LOAD PATH/<13><10>

238 PRINTNOCNTLR:   NOCNTLR_MESSAGE
                    .ASCIC  /NO SUPPORTED DISK CONTROLLER FOUND/<13><10>

239 PRINTNODRIVE01: NODRIVE01_MESSAGE
                    .ASCIC  /UNSUPPORTED DISK CONFIGURATION FOUND - DRIVE 0 AND 1/<13><10>

240 PRINTPROPER:    PROPER_MESSAGE
                    .ASCIC  <13><10><13><10>/**       PROPER SETUP REQUIRED          **/
```

```
                                                03DB   241 PRINTNSPN6025:  NSPN6025_MESSAGE
4B 43 45 48 43 20 20 2A 2A 0A 0D 00'            03DB                .ASCIC  <13><10>7**  CHECK FOR DIAGNOSTIC PACK CRDPACK  **/-
53 4F 4E 47 41 49 44 20 52 4F 46 20             03E7
44 52 43 20 4B 43 41 50 20 43 49 54             03F3
                  2A 2A 09 4B 43 41 50          03FF
45 53 4E 49 20 20 20 20 2A 2A 0A 0D             0406                <13><10>/**     INSERTED AND SPINNING IN /
49 50 53 20 44 4E 41 20 44 45 54 52             0412
         20 4E 49 20 47 4E 49 4E 4E             041E
                                 4B             03DB
                                                0427   242 PRINTNSPN8081:  NSPN8081_MESSAGE
48 43 20 20 20 20 20 2A 2A 0A 0D 00'            0427                .ASCIC  <13><10>7**      CHECK THAT DUA0: IS SPINNING      **/<13><10>
41 55 44 20 54 41 48 54 20 4B 43 45             0433
49 4E 4E 49 50 53 20 53 49 20 3A 30             043F
   0A 0D 2A 2A 20 20 20 20 20 47 4E             044B
                              2E                0427
                                                0456   243 PRINTSPNEND:    SPNEND_MESSAGE
         0A 0D 2A 2A 20 20 20 20 00'            0456                .ASCIC  /    **/<13><10>
                                 08             0456
                                                045F   244 PRINTAUTO:      AUTO_MESSAGE
                     20 4F 54 55 41 00'         045F                .ASCIC  /AUTO /
                                 05             045F
                                                0465   245 PRINTMENU:      MENU_MESSAGE
                     20 55 4E 45 4D 00'         0465                .ASCIC  /MENU /
                                 05             0465
                                                046B   246 PRINTPART:      PART_MESSAGE
      20 20 20 31 20 54 52 41 50 00'            046B                .ASCIC  /PART 1   /
                                 09             046B
                                                0475   247 PRINTTEST:      TEST_MESSAGE
41 54 53 20 47 4E 49 54 53 45 54 00'            0475                .ASCIC  /TESTING STARTED - RUN TIME = 0:30 MINUTES/
54 20 4E 55 52 20 2D 20 44 45 54 52             0481
4D 20 30 33 3A 30 20 3D 20 45 4D 49             048D
                     53 45 54 55 4E 49          0499
                                 29             0475
                                                049F   248 PRINTCPU:       CPU_MESSAGE
   20 20 20 44 41 42 20 55 50 43 00'            049F                .ASCIC  /CPU BAD - /
                                 0A             049F
                                                04AA   249 PRINTRC25:      RC25_MESSAGE
                     20 35 32 43 52 00'         04AA                .ASCIC  /RC25 /
                                 05             04AA
                                                04B0   250 PRINTDAA0:      DAA0_MESSAGE
                     20 30 41 41 44 00'         04B0                .ASCIC  /DAA0 /
                                 05             04B0
                                                04B6   251 PRINTDUA0:      DUA0_MESSAGE
                     20 30 41 55 44 00'         04B6                .ASCIC  /DUA0 /
                                 05             04B6
                                                04BC   252 PRINTDJA0:      DJA0_MESSAGE
                     20 30 41 4A 44 00'         04BC                .ASCIC  /DJA0 /
                                 05             04BC
                                                04C2   253 PRINTDJA1:      DJA1_MESSAGE
                     20 31 41 4A 44 00'         04C2                .ASCIC  /DJA1 /
                                 05             04C2
                                                04C8   254 PRINTRA60:      RA60_MESSAGE
                     20 30 36 41 52 00'         04C8                .ASCIC  /RA60 /
                                 05             04C8
                                                04CE   255 PRINTRA80:      RA80_MESSAGE
                     20 30 38 41 52 00'         04CE                .ASCIC  /RA80 /
                                 05             04CE
```

```
                        04D4   256 PRINTRA81:      RA81_MESSAGE
                20 31 38 41 52 00' 04D4                .ASCIC  /RA8T /
                             05    04D4
                                   04DA   257
                        0000055E   04DA   258 BOO$GT_COMBUF:  .BLKB 132
                        000005E2   055E   259 BOO$GL_FREEMEM: .BLKB 132
                                   05E2   260
                                   05E2   261                 .ALIGN  PAGE            ; Align SCB on page boundary
                                   0600   262
                                   0600   263
                                   0600   264 SCB::                                   ; SYSTEM CONTROL BLOCK
                        00000000   0600   265 SCB0::           .LONG   0               ;       (RESERVED)
                        00000000   0604   266 SCB4::           .LONG   0               ;       MACHINE CHECK
                        00000620   0608   267 SCB8::           .BLKL   6               ;
                        00000000   0620   268 SCB20::          .LONG   0               ;       ACCESS CONTROL VIOLATION
                        00000660   0624   269 SCB24::          .BLKL   15              ;
                        00000000   0660   270 SCB60::          .LONG   0               ;       WRITE NON-EXISTENT MEMORY
                        00000800   0664   271 SCB64::          .BLKL   103             ;
                                   0800   272
                                   0800   273
                                   0800   274                 .ALIGN  PAGE            ; Align INTERRUPT STACK
                                   0800   275                                         ; on page boundary
                        000009FC   0800   276 KERNEL_STACK::  .BLKL   127             ; Allocate a stack area
                        00000000   09FC   277 TOP_OF_KSTACK:: .LONG   0               ; Set top of stack pointer
                                   0A00   278
                                   0A00   279                 .ALIGN  PAGE            ; Align INTERRUPT STACK
                                   0A00   280                                         ; on page boundary
                        00000BFC   0A00   281 INTERRUPT_STACK:: .BLKL  127             ; Allocate a stack area
                        00000000   0BFC   282 TOP_OF_ISTACK:: .LONG   0               ; Set top of stack pointer
                                   0C00   283
                                   0C00   284                 .ALIGN  PAGE            ; Align RPB on page boundary
                                   0C00   285
                        00000E00   0C00   286 RPB_BUFF::      .BLKL   128             ; Set top of stack pointer
                                   0E00   287
                                   0E00   288 ;++
                                   0E00   289 ;
                                   0E00   290 ; Facility:  System Bootstrapping
                                   0E00   291 ;
                                   0E00   292 ; Abstract: CONIO provides basic console read,readprompt and write facilities.
                                   0E00   293 ;
                                   0E00   294 ; Environment: Mode=Kernel
                                   0E00   295 ;
                                   0E00   296 ; Author:  RICHARD I. HUSTVEDT, Creation date:  27-APR-1978
                                   0E00   297 ;
                                   0E00   298 ; Modified by: PETER GREEN      23-MAY-1983
                                   0E00   299 ;
                                   0E00   300 ;       MODIFIED TO PROMPT FOR ERROR PREPARATION MESSAGES.
                                   0E00   301 ;
                                   0E00   302 ;       Darryl L. Ball  14-June-1984
                                   0E00   303 ;
                                   0E00   304 ;       Added 730/725 Processor register definitions. Needed for compilation
                                   0E00   305 ;       under Version 4.X of VAX/VMS.
                                   0E00   306 ;
                                   0E00   307 ;
                                   0E00   308 ;--
                                   0E00   309
                                   0E00   310 ;
```

```
                        0E00    311 ; Include files:
                        0E00    312 ;
                        0E00    313         $PRDEF                          ; Define processor registers
                        0E00                .SAVE   LOCAL_BLOCK
                        0E00                .PSECT  $ABS$,ABS
                 00000E00                   .RESTORE
                        0E00    314         $PR730DEF                       ; Define 730/725 processor registers
                        0E00                .SAVE   LOCAL_BLOCK
                        0E00                .PSECT  $ABS$,ABS
                 00000E00                   .RESTORE
                        0E00    315         $SSDEF                          ; Define status code values
                        0E00                .SAVE   LOCAL_BLOCK
                        0E00                .PSECT  $ABS$,ABS
                 00000E00                   .RESTORE
                        0E00    316
                        0E00    317 ;
                        0E00    318 ; MACROS:
                        0E00    319 ;
                        0E00    320
                        0E00    321 ;
                        0E00    322 ; Equated Symbols:
                        0E00    323 ;
                        0E00    324 ;
         0000000D       0E00    325         CR      = 13                    ; Character code for carriage return
         0000000A       0E00    326         LF      = 10                    ; Character code for line feed
         00000015       0E00    327         CONTROL_U = 21                  ; Character code for control-U
         0000007F       0E00    328         RUBOUT  = 127                   ; Character code for rubout
         00000000       0E00    329         V_RUB   = 0                     ; Rubout sequence in progress
                        0E00    330
                        0E00    331 ;
                        0E00    332 ; Own Storage:
                        0E00    333 ;
```

```
                        0E00   335          .SBTTL  BOO$READPROMPT - Prompt and read input string
                        0E00   336 ;++
                        0E00   337 ; Functional Description:
                        0E00   338 ;
                        0E00   339 ;       BOO$READPROMPT outputs the specified ASCIZ prompt string on the
                        0E00   340 ;       console terminal then checks the count of characters to be read.
                        0E00   341 ;       If zero it exits, otherwise it reads the console terminal until
                        0E00   342 ;       either a carriage return is encountered or the character count
                        0E00   343 ;       is satisfied.  The specified buffer is filled with an ASCIC
                        0E00   344 ;       string containing the characters read but not including the
                        0E00   345 ;       terminating carriage return.
                        0E00   346 ;
                        0E00   347 ; Calling Sequence:
                        0E00   348 ;       CALLG   ARGLIST,BOO$READPROMPT
                        0E00   349 ;
                        0E00   350 ; Input Parameters:
                        0E00   351 ;       PROMPT(AP)  -  Address of ASCIZ prompt string
 00000004               0E00   352          PROMPT  =  4
                        0E00   353 ;
                        0E00   354 ;       SIZE(AP)    -  Maximum length of input string
 00000008               0E00   355          SIZE    =  8
                        0E00   356 ;                        Note: if size is zero, then nothing is read
                        0E00   357 ;                              and only the prompt string is written.
                        0E00   358 ;
                        0E00   359 ;       BUF(AP)     -  Address of input buffer
 0000000C               0E00   360          BUF     =  12
                        0E00   361 ;
                        0E00   362 ; Output Parameters:
                        0E00   363 ;       R0 - Completion status code (always SS$_NORMAL)
                        0E00   364 ;
                        0E00   365 ;       Buffer located by BUF(AP) will be filled with the string
                        0E00   366 ;       read as an ASCIC string.
                        0E00   367 ;
                        0E00   368 ;--
                        0E00   369          .LIST   MEB                     ; Show macro expansions
                        0E00   370 BOO$READPROMPT::                         ;
                  0394  0E00   371          .WORD   ^M<R2,R4,R7,R8,R9>      ;
58   04 AC      D0     0E02   372 5$:       MOVL    PROMPT(AP),R8           ; Point to prompt string
     0089        30     0E06   373          BSBW    OUTZSTR                 ; And send it to the console
52   08 AC      D0     0E09   374          MOVL    SIZE(AP),R2             ; Maximum number of characters to read
        6B      13     0E0D   375          BEQL    70$                     ; Done if nothing to read
        54      D4     0E0F   376          CLRL    R4                      ; Clear flags
59   0C AC      D0     0E11   377          MOVL    BUF(AP),R9              ; Set address of input buffer
        89      94     0E15   378          CLRB    (R9)+                   ; Initialize string count
   57   69      9E     0E17   379          MOVAB   (R9),R7                 ; Remember start of buffer
     07 52      F5     0E1A   380          SOBGTR  R2,10$                  ; Decrement and test character count
   50   0D      9A     0E1D   381          MOVZBL  #CR,R0                  ; Set to send CR/LF
        65      10     0E20   382          BSBB    OUTCHAR                 ; Send CR
        49      11     0E22   383          BRB     60$                     ; End of read
   50   20      DB     0E24   384 10$:      MFPR    #PR$_RXCS,R0            ; Get console status
F9 50   07      E1     0E27   385          BBC     #7,R0,10$               ; Wait for ready
   58   21      DB     0E2B   386          MFPR    #PR$_RXDB,R8            ; Get console character
58   80 8F      8A     0E2E   387          BICB    #^X80,R8                ; Strip parity
7F 8F   58      91     0E32   388          CMPB    R8,#RUBOUT              ; Check for rubout
        1F      12     0E36   389          BNEQ    40$                     ; No
02 54   00      E2     0E38   390          BBSS    #V_RUB,R4,20$           ; Set start of rubout sequence
        40      10     0E3C   391          BSBB    OUTBSLSH                ; Output back slash
```

```
                58   79   9A  0E3E    392 20$:    MOVZBL  -(R9),R8             ; Get rubbed out character
                     08   12  0E41    393         BNEQ    30$                 ; Skip increment
                     59   D6  0E43    394         INCL    R9                  ; Point at start of buffer
             52   08  AC   D0  0E45    395         MOVL    SIZE(AP),R2         ; Restore size limit
                     D9   11  0E49    396         BRB     10$                 ; And get another character
                     37   10  0E4B    397 30$:    BSBB    OUTR8               ; Output rubbed out character
                     52   D7  0E4D    398         DECL    R2                  ; One less character
                     D3   11  0E4F    399         BRB     10$                 ; And get another
             02 54   00   E5  0E51    400         BBCC    #V_RUB,R4,40$       ; Terminate rubout sequence
                     27   10  0E55    401         BSBB    OUTBSLSH            ; Output backslash
                58   15   91  0E57    402 40$:    CMPB    #CONTROL_U,R8       ; Check for line cancel (control-U)
                     A6   13  0E5A    403         BEQL    5$                  ; Branch if so
             03 58   06   E1  0E5C    404         BBC     #6,R8,50$           ; Branch if not alpha
                58   20   8A  0E60    405         BICB    #32,R8              ; Set to upper case
                                0E63    406 50$:                            ;
                     1F   10  0E63    407         BSBB    OUTR8               ; Echo character
                89   58   90  0E65    408         MOVB    R8,(R9)+            ; Buffer new character
                58   0D   91  0E68    409         CMPB    #CR,R8              ; Is character a carriage return?
                     B7   12  0E6B    410         BNEQ    10$                 ; No,
                50   0A   9A  0E6D    411 60$:    MOVZBL  #LF,R0              ; Yes send line feed also
                     15   10  0E70    412         BSBB    OUTCHAR             ; Output character in R0
                59   57   C2  0E72    413         SUBL    R7,R9               ; Compute character count
                     59   D7  0E75    414         DECL    R9                  ; Remove carriage return
                77   59   90  0E77    415         MOVB    R9,-(R7)            ; and form counted string
                50   01   3C  0E7A    416 70$:    MOVZWL  #SS$_NORMAL,R0      ; Return normal completion status
                               04  0E7D    417         RET                 ;
                                0E7E    418 OUTBSLSH:                       ; Output back slash
             50    5C   8F   9A  0E7E    419         MOVZBL  #^A/\/,R0           ; Set character code
                     03   11  0E82    420         BRB     OUTCHAR             ; And output it
                50   58   9A  0E84    421 OUTR8:  MOVZBL  R8,R0               ; Get character to output
                               0E87    422 OUTCHAR:                        ; Output character in R0
                51   22   DB  0E87    423         MFPR    #PR$_TXCS,R1        ; Get console transmitter status
             F9 51   07   E1  0E8A    424         BBC     #7,R1,OUTCHAR       ; Wait for ready
                23   50   DA  0E8E    425         MTPR    R0,#PR$_TXDB        ; Send character to console
                               05  0E91    426 OUTEX:  RSB                 ; Return
                                0E92    427 OUTZSTR:                        ; Output ASCIZ string
                50   B8   9A  0E92    428         MOVZBL  (R8)+,R0            ; Get next character to output
                     FA   13  0E95    429         BEQL    OUTEX               ; Branch if at end of string
                     EE   10  0E97    430         BSBB    OUTCHAR             ; Send character
                     F7   11  0E99    431         BRB     OUTZSTR             ; Continue for all characters
                                0E9B    432                                 ;
                                0E9B    433                                 ;
```

```
                                0E9B   435          .SBTTL  PROMPT1 - Routine to prompt console for input
                                0E9B   436 ;
                                0E9B   437 ; Calling Sequence:
                                0E9B   438 ;          BSBW    PROMPT1
                                0E9B   439 ;          .ASCIZ  /prompt string/
                                0E9B   440 ;
                                0E9B   441 ; Ouput Parameters:
                                0E9B   442 ;          COMBUF - Filled with ASCIC string representing reply.
                                0E9B   443 ;                   Reply does not include terminator.
                                0E9B   444 ;
                                0E9B   445 ;
                                0E9B   446 PROMPT1:                                                    ;
                      50   6E   D0  0E9B   447          MOVL    (SP),R0                     ; Get prompt string pointer
                    F638   CF   9F  0E9E   448          PUSHAB  W^BOO$GT_COMBUF             ; Set buffer address
                 7E   84   8F   9A  0EA2   449          MOVZBL  #BOO$C_COMBUFSZ,-(SP)       ;  and size for read.
                           50   DD  0EA6   450          PUSHL   R0                          ; Set address of prompt string
                FF53   CF   03   FB  0EA8   451          CALLS   #3,W^BOO$READPROMPT         ; Prompt and read response
             CO BE   0100   8F   00   3A  0EAD   452          LOCC    #0,#256,@(SP)               ; Find end of string
                       01   A1   17  0EB4   453          JMP     1(R1)                       ; and return
                                0EB7   454
                                0EB7   455
                                0EB7   456          .SBTTL  BOO$TYPE_ASCIC - Type a counted ASCII string
                                0EB7   457 ;++
                                0EB7   458 ; Functional Description:
                                0EB7   459 ;          This routine accepts a string descriptor and types the message
                                0EB7   460 ;          on the console terminal
                                0EB7   461 ;
                                0EB7   462 ; Calling Sequence:
                                0EB7   463 ;          BSBW    BOO$TYPE_ASCIC
                                0EB7   464 ;
                                0EB7   465 ; Inputs:
                                0EB7   466 ;          R1 = Address of ASCIC string
                                0EB7   467 ;
                                0EB7   468 ; Outputs:
                                0EB7   469 ;          R0,R1 altered
                                0EB7   470 ;          All other registers preserved
                                0EB7   471 ;--
                                0EB7   472
                                0EB7   473 BOO$TYPE_ASCIC:
                      50   81   9A  0EB7   474          MOVZBL  (R1)+,R0                    ; R0 = Size, R1 = Address of string
                           3C   BB  0EBA   475          PUSHR   #^M<R2,R3,R4,R5>            ; Save registers
                53   F69E   CF   D0  0EBC   476          MOVL    W^BOO$GL_FREEMEM,R3         ; Address of scratch storage
                           7E   7C  0EC1   477          CLRQ    -(SP)                       ; No read for PROMPTREAD call
                           53   DD  0EC3   478          PUSHL   R3                          ; Address of ASCIZ string
                63   61   50   28  0EC5   479          MOVC3   R0,(R1),(R3)                ; Move the string to scratch storage
                           83   94  0EC9   480          CLRB    (R3)+                       ; and make it ASCIZ
                FF30   CF   05   FB  0ECB   481          CALLS   #3,W^BOO$READPROMPT         ; Type the string
                           3C   8A  0ED0   482          POPR    #^M<R2,R3,R4,R5>            ; Recover saved registers
                                05  0ED2   483          RSB
                                0ED3   484 ;
                                0ED3   485 ; THIS ROUTINE PRINTS THE MESSSAGE SHOWN AND WAITS FOR USER
                                0ED3   486 ; INTERVENTION ,<CR>, FOR PREPARATION ERRORS.
                                0ED3   487 ;
                                0ED3   488 PROMPT_MESS:
                                   C6   10  0ED3   489          BSBB    PROMPT1                     ; NO, PROMPT TO SPIN
20 4F 45 48 54 20 20 20 20 20 2A 2A  0ED5   490 .ASCIZ  /**     THEN TYPE <CR> TO CONTINUE      **/
4F 54 20 3E 52 43 3C 20 45 50 59 54  0EF1
```

```
2A 2A 09 45 55 4E 49 54 4E 4F 43 20   0EED
                                 00   0EF9
                  51    FSDC CF      9E   0EFA   491          MOVAB   BOO$G1_COMBUF,R1
                                B6    10   0EFF   492          BSBB    BOO$TYPE_ASCIC
                   00000000'EF        94   0F01   493          CLRB    DS$GB_UDA50_INIT
              00000000'EF    00       FB   0F07   494          CALLS   #0,UD_DRIVER1
                                           0F0E   495
```

```
                        0F0E   497 .SBTTL PRINT ROUTINE
                        0F0E   498
                        0F0E   499 ;++
                        0F0E   500 ;  FUNCTIONAL DESCRIPTION:
                        0F0E   501 ;
                        0F0E   502 ; THIS ROUTINE IS USED TO TRANSMIT AN ASCIC MESSAGE TO THE CONSOLE
                        0F0E   503 ; TERMINAL.
                        0F0E   504 ;--
                        0F0E   505 PRINT_ROUTINE:
      51    01    D0    0F0E   506          MOVL    #1,R1                   ; Initialize index
      53    64    9A    0F11   507          MOVZBL  (R4),R3                 ; Get number of chars to print
            52    D4    0F14   508          CLRL    R2                      ; Clear character stuffer
   52    6441    9A    0F16   509 10$:      MOVZBL  (R4)[R1],R2             ; Get a character
      50    22    DB    0F1A   510 20$:      MFPR    #TXCS,R0                ; Get tx status
   F9 50    07    E1    0F1D   511          BBC     #7,R0,20$               ; Transmitter ready?
      23    52    DA    0F21   512          MTPR    R2,#TXDB                ; Print a character
   EE 51    53    F3    0F24   513          AOBLEQ  R3,R1,10$               ; Print the whole message
                  05    0F28   514          RSB                             ; EXIT
                        0F29   515
```

```
                        0F29    517 .SBTTL ERROR ROUTINE
                        0F29    518 ;++
                        0F29    519 ; FUNCTIONAL DESCRIPTION:
                        0F29    520 ;
                        0F29    521 ; THIS ROUTINE IS USED TO DISPLAY AN ERROR MESSAGE IN CASE OF CPU OR
                        0F29    522 ; LOAD PATH FAILURE OR WHEN WARNING MESSAGE WHEN NO CRD SUPPORTED
                        0F29    523 ; DISK CONTROLER IS FOUND. IT ALSO ABORTS AFTER ALL MESSAGES ARE PRINTED.
                        0F29    524 ;
                        0F29    525 ;                   R4 = ADDRESS OF ASCII MESSAGE
                        0F29    526 ;--
                        0F29    527
                        0F29    528 ERROR_ROUTINE::
      12 F2D6 CF    E8  0F29    529         BLBS    WHAT_MODE,10$           ; DETERMINE WHAT MODE
                        0F2E    530                                         ; AUTO OR MENU
54    F313 CF       DE  0F2E    531         MOVAL   PRINTFAIL,R4            ;     FAILED
        D8 AF       16  0F33    532         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
54    F525 CF       DE  0F36    533         MOVAL   PRINTAUTO,R4            ;    AUTO
        D0 AF       16  0F3B    534         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
              08    11  0F3E    535         BRB     20$
54    F521 CF       DE  0F40    536 10$:    MOVAL   PRINTMENU,R4            ;    MENU
        C6 AF       16  0F45    537         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                        0F48    538
54    F308 CF       DE  0F48    539 20$:    MOVAL   PRINTABORT,R4           ;     FAILED
        BE AF       16  0F4D    540         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                        0F50    541
      15 F283 CF    E8  0F50    542         BLBS    NOCNTLR_FND,NO_LOAD     ; IS THERE A CRD SUPPORTED
                        0F55    543                                         ; CONTROLLER?
18 00000000'EF      E8  0F55    544         BLBS    NODRIVE_0_1,DRIVE_NFND  ; IS THERE A CRD SUPPORTED
                        0F5C    545                                         ; DISK DRIVE 0 OR 1?
      1D F2AB CF    E8  0F5C    546         BLBS    LOAD_ERROR,LOAD_BAD     ; IS THERE A CRD SUPPORTED
                        0F61    547
20 00000000'EF      E8  0F61    548         BLBS    BAD_DISK,DISK_ERR       ; BAD STATUS FROM ONLINE COMMAND
                        0F68    549
              29    11  0F68    550         BRB     CPU_ERR                 ; CPU ERROR
                        0F6A    551 NO_LOAD::
54    F3E3 CF       DE  0F6A    552         MOVAL   PRINTNOCNTLR,R4         ; NO CRD SUPPORTED CONTROLLER FOUND
        9C AF       16  0F6F    553         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
              28    11  0F72    554         BRB     ERR_END
                        0F74    555
                        0F74    556 DRIVE_NFND:
54    F3FE CF       DE  0F74    557         MOVAL   PRINTNODRIVE01,R4       ; NO CRD SUPPORTED DISK DRIVE
                        0F79    558                                         ; 0 OR 1 FOUND
        92 AF       16  0F79    559         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
              1E    11  0F7C    560         BRB     ERR_END
                        0F7E    561
                        0F7E    562 LOAD_BAD:
54    F39A CF       DE  0F7E    563         MOVAL   PRINTLOADMCK,R4         ; UNEXPECTED MACHINE CHECK WHILE
                        0F83    564                                         ; VERIFYING LOAD PATH
        88 AF       16  0F83    565         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
              14    11  0F86    566         BRB     ERR_END
                        0F88    567 DISK_ERR:
54    F383 CF       DE  0F88    568         MOVAL   PRINTDISKERR,R4         ; UNEXPECTED MACHINE CHECK WHILE
                        0F8D    569                                         ; VERIFYING LOAD PATH
        FF7D CF     16  0F8D    570         JSB     PRINT_ROUTINE           ; PRINT MESSAGE
              09    11  0F91    571         BRB     ERR_END
                        0F93    572 CPU_ERR:
54    F508 CF       DE  0F93    573         MOVAL   PRINTCPU,R4             ;     CPU BAD
```

```
                FF72 CF    16  0F98    574          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0F9C    575
                           0F9C    576 ERR_END:
          54    F292 CF    DE  0F9C    577          MOVAL   PRINTFIELD,R4           ; CALL FIELD SERVICE
                FF69 CF    16  0FA1    578          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0FA5    579
          54    F2BE CF    DE  0FA5    580          MOVAL   PRINTEND,R4             ; END OF VAX-11/730
                FF60 CF    16  0FAA    581          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0FAE    582
            0B  F251 CF    E8  0FAE    583          BLBS    WHAT_MODE,30$           ; DETERMINE WHAT MODE
                           0FB3    584                                              ; AUTO OR MENU
          54    F4A8 CF    DE  0FB3    585          MOVAL   PRINTAUTO,R4            ;    AUTO
                FF52 CF    16  0FB8    586          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                     09    11  0FBC    587          BRB     40$
                           0FBE    588
          54    F4A3 CF    DE  0FBE    589 30$:     MOVAL   PRINTMENU,R4            ;    MENU
                FF47 CF    16  0FC3    590          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0FC7    591
          54    F2BA CF    DE  0FC7    592 40$:     MOVAL   PRINTTEND,R4            ;    TEST **
                FF3E CF    16  0FCC    593          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0FD0    594
          54    F2BF CF    DE  0FD0    595          MOVAL   PRINTRET,R4             ; RETURNING TO CONSOLE PROMPT
                FF35 CF    16  0FD5    596          JSB     PRINT_ROUTINE           ; PRINT MESSAGE
                           0FD9    597
    00000174'EF    16  0FD9    598          JSB     CLEAN_UP                ; Go to Clean up routine
                     00    0FDF    599          HALT                            ; Halt the processor
                           0FE0    600
```

```
                        0FE0    602 .SBTTL  RA60 PRINT ROUTINE
                        0FE0    603 ;++
                        0FE0    604 ; FUNCTIONAL DESCRIPTION:
                        0FE0    605 ;
                        0FE0    606 ; This routine is used to display a message when RA60 CRD supported
                        0FE0    607 ; disk controller is found. It also CHECKS WHAT DRIVE AND IF THE DRIVE IS
                        0FE0    608 ; SPINNING and CORRECT PACK LABEL.
                        0FE0    609 ;--
                        0FE0    610
                        0FE0    611 RA60_TEST::
   48 F21F CF    E8     0FF0    612             BLBS    WHAT_MODE,30$
   43 F22E CF    E8     0FE5    613             BLBS    RA60_MESS_ST,30$
   54    F235 CF    DE  0FEA    614             MOVAL   PRINTPASS,R4            ; PRINT PASSED TO CPU PART 3
         FF1B CF    16  0FEF    615             JSB     PRINT_ROUTINE           ; PRINT MESSAGE
         F220 CF    D6  0FF3    616             INCL    RA60_MESS_ST
   54    F4CD CF    DE  0FF7    617             MOVAL   PRINTRA60,R4            ; PRINT MESSAGE
         FF0E CF    16  0FFC    618             JSB     PRINT_ROUTINE           ; RA60
   54    F467 CF    DE  1000    619             MOVAL   PRINTPART,R4            ; PRINT MESSAGE
         FF05 CF    16  1005    620             JSB     PRINT_ROUTINE           ; PART  1
0B 00000000'EF    E9  1009    621             BLBC    WHAT_DRIVE,10$          ; WHAT DRIVE?
   54    F4AE CF    DE  1010    622             MOVAL   PRINTDJA1,R4            ; PRINT MESSAGE
         FEF5 CF    16  1015    623             JSB     PRINT_ROUTINE           ; DJA1
               09   11  1019    624             BRB     20$
   54    F49D CF    DE  101B    625 10$:        MOVAL   PRINTDJA0,R4            ; PRINT MESSAGE
         FEEA CF    16  1020    626             JSB     PRINT_ROUTINE           ; DJA0
   54    F44D CF    DE  1024    627 20$:        MOVAL   PRINTTEST,R4            ; PRINT MESSAGE
         FEE1 CF    16  1029    628             JSB     PRINT_ROUTINE           ; TESTING STARTED AND TIME
04 00000000'EF    E9  102D    629 30$:        BLBC    BAD_DISK,40$
         FEF1 CF    16  1034    630             JSB     ERROR_ROUTINE
03 00000000'EF    E8  1038    631 40$:        BLBS    SPIN_DWN,50$            ; IS IT SPININNING
               EFBE' 31  103F    632             BRW     TEST_8                  ; YES READ PACK LABEL
   54    F367 CF    DE  1042    633 50$:        MOVAL   PRINTPROPER,R4          ; PRINT MESSAGE
         FEC3 CF    16  1047    634             JSB     PRINT_ROUTINE           ; PROPER SETUP REQUIRED
   54    F38C CF    DE  104B    635             MOVAL   PRINTNSPN6025,R4        ; PRINT MESSAGE
         FEBA CF    16  1050    636             JSB     PRINT_ROUTINE           ; CHECK CRDPACK AND SPINNING
                        1054    637
0B 00000000'EF    E9  1054    638             BLBC    WHAT_DRIVE,60$
                        105B    639
   54    F463 CF    DE  105B    640             MOVAL   PRINTDJA1,R4            ; PRINT MESSAGE
         FEAA CF    16  1060    641             JSB     PRINT_ROUTINE           ; DJA1
               09   11  1064    642             BRB     70$
   54    F452 CF    DE  1066    643 60$:        MOVAL   PRINTDJA0,R4            ; PRINT MESSAGE
         FE9F CF    16  106B    644             JSB     PRINT_ROUTINE           ; DJA0
   54    F3E3 CF    DE  106F    645 70$:        MOVAL   PRINTSPNEND,R4          ; PRINT MESSAGE
         FE96 CF    16  1074    646             JSB     PRINT_ROUTINE           ;     **
1B 00000000'EF    E9  1078    647             BLBC    HMBLK_ERR,80$           ; HOME BLOCK ERROR?
   54    F24D CF    DE  107F    648             MOVAL   PRINTDRVERR,R4          ; PRINT MESSAGE
         FE86 CF    16  1084    649             JSB     PRINT_ROUTINE           ; IF SETUP CORRECT SUSPECT DRIVE ERR
   54    F1A6 CF    DE  1088    650             MOVAL   PRINTFIELD,R4           ; PRINT MESSAGE
         FE7D CF    16  108D    651             JSB     PRINT_ROUTINE           ; CALL FIELD SERVICE
   54    F3C1 CF    DE  1091    652             MOVAL   PRINTSPNEND,R4          ; PRINT MESSAGE
         FE74 CF    16  1096    653             JSB     PRINT_ROUTINE           ;     **
         FE35 CF    16  109A    654 80$:        JSB     PROMPT_MESS             ; PROMPT FOR SETUP
                        109E    655
```

```
                                109E    657 .SBTTL  RA80 AND RA81 PRINT ROUTINE
                                109E    658 ;++
                                109E    659 ; FUNCTIONAL DESCRIPTION:
                                109E    660 ;
                                109E    661 ; This routine is used to display a message when AN RA80 OR RA81 CRD
                                109E    662 ; supported disk configuration is found, (in drive 0). It also CHECKS for
                                109E    663 ; DRIVE 0 AND IF THE DRIVE IS SPINNING.
                                109E    664 ;
                                109E    665 ;--
                                109E    666 RA8081_TEST::
03 00000000'EF    E9            109E    667             BLBC    WHAT_DRIVE,10$          ; IF DRIVE 1
          EF58'   31            10A5    668             BRW     INIT_DRV0               ; CHECK DRIVE 0
    48 F157 CF    E8            10A8    669 10$:        BLBS    WHAT_MODE,40$
    43 F16A CF    E8            10AD    670             BLBS    RA8081_MESS_ST,40$
 54   F16D CF    DE            10B2    671             MOVAL   PRINTPASS,R4            ; PRINT PASSED
      FE53 CF    16            10B7    672             JSB     PRINT_ROUTINE           ; PRINT MESSAGE
      F15C CF    D6            10BB    673             INCL    RA8081_MESS_ST
08 00000000'EF    E9            10BF    674             BLBC    RA81_PRES,20$           ; RA81?
                                10C6    675                                             ; IF 0 THEN RA80
 54   F40A CF    DE            10C6    676             MOVAL   PRINTRA81,R4            ; PRINT MESSAGE
      FE3F CF    16            10CB    677             JSB     PRINT_ROUTINE           ; RA81
           09    11            10CF    678             BRB     30$                     ; CONTINUE TO PRINT MESSAGE
                                10D1    679                                             ; TEST STARTED
 54   F3F9 CF    DE            10D1    680 20$:        MOVAL   PRINTRA80,R4            ; PRINT MESSAGE
      FE34 CF    16            10D6    681             JSB     PRINT_ROUTINE           ; RA80
 54   F38D CF    DE            10DA    682 30$:        MOVAL   PRINTPART,R4            ; PRINT MESSAGE
      FE2B CF    16            10DF    683             JSB     PRINT_ROUTINE           ; PART  1
 54   F3CF CF    DE            10E3    684             MOVAL   PRINTDUA0,R4            ; PRINT MESSAGE
      FE22 CF    16            10E8    685             JSB     PRINT_ROUTINE           ; DUA0
 54   F385 CF    DE            10EC    686             MOVAL   PRINTTEST,R4            ; PRINT MESSAGE
      FE19 CF    16            10F1    687             JSB     PRINT_ROUTINE           ; TESTING STARTED AND TIME
04 00000000'EF    E9            10F5    688 40$:        BLBC    BAD_DISK,50$            ;
      FE29 CF    16            10FC    689             JSB     ERROR_ROUTINE           ;
 00000000'EF      95            1100    690 50$:        TSTB    SPIN_DWN                ; IS IT SPININNIG
           03    12            1106    691             BNEQ    60$
          EEF5'  31            1108    692             BRW     LESI_UDA_PARAM          ; YES BOOT SUPERVISOR PORTION
 54   F29E CF    DE            1108    693 60$:        MOVAL   PRINTPROPER,R4          ; PRINT MESSAGE
      FDFA CF    16            1110    694             JSB     PRINT_ROUTINE           ; PROPER SETUP REQUIRED
 54   F30F CF    DE            1114    695             MOVAL   PRINTNSPN8081,R4        ; PRINT MESSAGE
      FDF1 CF    16            1119    696             JSB     PRINT_ROUTINE           ; CHECK THAT DUA0 IS SPINNING
      FDB2 CF    16            111D    697             JSB     PROMPT_MESS             ; PROMPT FOR SETUP
                                1121    698
```

```
                        1121    700 .SBTTL  RC25 PRINT ROUTINE
                        1121    701 ;++
                        1121    702 ; FUNCTIONAL DESCRIPTION:
                        1121    703 ;
                        1121    704 ; This routine is used to display a message when RC25 CRD supported
                        1121    705 ; disk OPTION is found. It also CHECKS for DRIVE 0, IF THE DRIVE IS
                        1121    706 ; SPINNING and for correct pack label.
                        1121    707 ;--
                        1121    708 RC25_TEST::
03 00000000'EF   E9     1121    709             BLBC    WHAT_DRIVE,10$          ; IF DRIVE 1
         EED5'   31     1128    710             BRW     INIT_DRV0               ; CHECK DRIVE 0
   36 FOD4 CF   E8     1128    711 10$:        BLBS    WHAT_MODE,20$
   31 FOE6 CF   E8     1130    712             BLBS    RC25_MESS_ST,20$
   54  FOEA CF   DE     1135    713             MOVAL   PRINTPASS,R4            ; PRINT PASSED
       FDD0 CF   16     113A    714             JSB     PRINT_ROUTINE           ; PRINT MESSAGE
       FODD CF   D6     113E    715             INCL    RC25_MESS_ST
   54  F364 CF   DE     1142    716             MOVAL   PRINTRC25,R4            ; PRINT MESSAGE
       FDC3 CF   16     1147    717             JSB     PRINT_ROUTINE           ; RC25
   54  F31C CF   DE     114B    718             MOVAL   PRINTPART,R4            ; PRINT MESSAGE
       FDBA CF   16     1150    719             JSB     PRINT_ROUTINE           ; PART  1
   54  F358 CF   DE     1154    720             MOVAL   PRINTDAA0,R4            ; PRINT MESSAGE
       FDB1 CF   16     1159    721             JSB     PRINT_ROUTINE           ; DUA0
   54  F314 CF   DE     115D    722             MOVAL   PRINTTEST,R4            ; PRINT MESSAGE
       FDA8 CF   16     1162    723             JSB     PRINT_ROUTINE           ; TESTING STARTED AND TIME
04 00000000'EF   E9     1166    724 20$:        BLBC    BAD_DISK,30$            ; DRIVE ERROR?
       FDB8 CF   16     116D    725             JSB     ERROR_ROUTINE
03 00000000'EF   E8     1171    726 30$:        BLBS    SPIN_DWN,40$            ; IS IT SPININNIG
         EE85'   31     1178    727             BRW     TEST_8                  ; YES READ PACK LABEL
   54  F22E CF   DE     117B    728 40$:        MOVAL   PRINTPROPER,R4          ; PRINT MESSAGE
       FD8A CF   16     1180    729             JSB     PRINT_ROUTINE           ; PROPER SETUP REQUIRED
   54  F253 CF   DE     1184    730             MOVAL   PRINTNSPN6025,R4        ; PRINT MESSAGE
       FDR1 CF   16     1189    731             JSB     PRINT_ROUTINE           ; CHECK CRDPACK AND SPINNING
   54  F31F CF   DE     118D    732             MOVAL   PRINTDAA0,R4            ; PRINT MESSAGE
       FD78 CF   16     1192    733             JSB     PRINT_ROUTINE           ; DUA0
   54  F28C CF   DE     1196    734             MOVAL   PRINTSPNEND,R4          ; PRINT MESSAGE
       FD6F CF   16     1198    735             JSB     PRINT_ROUTINE           ;    **
1B 00000000'EF   E9     119F    736             BLBC    HMBLK_ERR,50$           ; HOME BLOCK ERROR?
   54  F126 CF   DE     11A6    737             MOVAL   PRINTDRVERR,R4          ; PRINT MESSAGE
       FD5F CF   16     11AB    738             JSB     PRINT_ROUTINE           ; IF SETUP CORRECT SUSPECT DRIVE ERR
   54  F07F CF   DE     11AF    739             MOVAL   PRINTFIELD,R4           ; PRINT MESSAGE
       FD56 CF   16     11B4    740             JSB     PRINT_ROUTINE           ; CALL FIELD SERVICE
   54  F29A CF   DE     11B8    741             MOVAL   PRINTSPNEND,R4          ; PRINT MESSAGE
       FD4D CF   16     11BD    742             JSB     PRINT_ROUTINE           ;    **
       FDOE CF   16     11C1    743 50$:        JSB     PROMPT_MESS             ; PROMPT FOR SETUP
                        11C5    744
```

```
                                11C5    746
                                11C5    747          .SBTTL  EXCEPTION HANDLER ROUTINES
                                11C5    748
                                11C5    749 .ALIGN LONG
                                11C8    750 ;++
                                11C8    751 ; FUNCTIONAL DESCRIPTION:
                                11C8    752 ;
                                11C8    753 ; This routine is used to handle all unexpected exceptions or
                                11C8    754 ; urgent interrupts (over priority ^X18). The results of an
                                11C8    755 ; unexpected exception or urgent interrupt is printing an error
                                11C8    756 ; message pointing to the CPU and aborting CRD Auto or Menu mode.
                                11C8    757 ;--
                                11C8    758 UNEX_E_I::
 26   FFFFFFFF  8F    DA         11C8    759          MTPR    #-1,#PR$_MCESR          ; CLEAR REGISTER
           5E    8E    C0       11CF    760          ADDL2   (SP)+,SP                ; ADJUST SP
F03A CF    0064  8F    B1       11D2    761          CMPW    #100,PASS_COUNT         ; TST IF IN HARDCORE
                       11    18 11D9    762          BGEQ    10$                     ; OR LOAD PATH SECTION
     F02F  CF    01    D0       11DB    763          MOVL    #1,CPU_BAD
     6E    FD45  CF    9E       11E0    764          MOVAB   ERROR_ROUTINE,(SP)      ; PRINT ERROR
                       0A    11 11E5    765          BRB     20$                     ; THEN ABORT
     F01F  CF    01    D0       11E7    766          MOVL    #1,LOAD_ERROR
     6E    FD39  CF    9E       11EC    767 10$:     MOVAB   ERROR_ROUTINE,(SP)
                       02    02 11F1    768 20$:     REI
                                11F2    769
                                11F2    770
                                11F2    771 .ALIGN LONG
                                11F4    772 ;++
                                11F4    773 ; This routine is used to handle expected MACHINE CHECKS
                                11F4    774 ; when an IDC is not found. The results of this expected
                                11F4    775 ; MACHINE CHECK is to continue and look for a LESI or
                                11F4    776 ; UDA50 controller on the UNIBUS.
                                11F4    777 ;--
                                11F4    778 MCHK_IDC::
 26   FFFFFFFF  8F    DA         11F4    779          MTPR    #-1,#PR$_MCESR          ; CLEAR REGISTER
           5E    8E    C0       11FB    780          ADDL2   (SP)+,SP                ; ADJUST SP
6E   00000000'EF    DE         11FE    781          MOVAL   UDA_LESI,(SP)           ; CONTINUE TO
                                1205    782                                          ; SIZE
                       02    02 1205    783          REI
                                1206    784
                                1206    785 .ALIGN LONG
                                1208    786 ;++
                                1208    787 ; This routine is used to handle expected MACHINE CHECKS
                                1208    788 ; when a LESI or UDA50 is not found. The results of this expected
                                1208    789 ; MACHINE CHECK is to print a warning message due to the lack of
                                1208    790 ; a supported disk load device and then abort.
                                1208    791 ;--
                                1208    792 MCHK_LES_UDA::
 26   FFFFFFFF  8F    DA         1208    793          MTPR    #-1,#PR$_MCESR          ; CLEAR REGISTER
           5E    8E    C0       120F    794          ADDL2   (SP)+,SP                ; ADJUST SP
     FFF0  CF    01    D0       1212    795          MOVL    #1,NOCNTLR_FND
     6E    FD0E  CF    9E       1217    796          MOVAB   ERROR_ROUTINE,(SP)      ; PRINT WARNING
                                121C    797                                          ; THEN ABORT
                       02    02 121C    798          REI
                                121D    799
                                121D    800 .ALIGN  LONG                ; A4
```

```
                                1220    802
                                1220    803             .SBTTL  INITIALIZATION ROUTINE
                                1220    804 ;++
                                1220    805 ; FUNCTIONAL DESCRIPTION:
                                1220    806 ;
                                1220    807 ;       This routine initializes the stack pointer, turns off
                                1220    808 ;       memory management and the stack pointers are invoked.
                                1220    809 ;       This routine also sets up the SCB buffer to refer to an unexpected
                                1220    810 ;       exception and urgent interrupt routine, moves to the
                                1220    811 ;       KERNEL stack and sets the IPL level to ^X18.
                                1220    812 ;
                                1220    813 ;       Assumptions:
                                1220    814 ;
                                1220    815 ;               1) Init was performed in the Console mode prior to
                                1220    816 ;               running this program.
                                1220    817 ;
                                1220    818 ;               Therefore :
                                1220    819 ;
                                1220    820 ;                       MODE  = KERNEL MODE
                                1220    821 ;                       IPL   = 31
                                1220    822 ;                       STACK = INTERRUPT STACK
                                1220    823 ;
                                1220    824 ;--
                                1220    825
                                1220    826
                  38    00  DA  1220    827 INIT1:  MTPR    #0,#PR$_MAPEN           ; Turn off memory management
                                1223    828
                EFDB  CF    55  D0  1223    829         MOVL    R5,WHAT_MODE            ; AUTO OR MENU/ R5 PASSED FROM
                                1228    830                                         ; CRAB00.CMD AND CRMB00.CMD
                                1228    831
                5E    F9D0  CF  DE  1228    832         MOVAL   TOP_OF_ISTACK,SP        ; SP Points to the top of stack
                      04    5E  DA  122D    833         MTPR    SP,#PR$_ISP             ; Load the ISP
                                1230    834
                5E    F7C8  CF  DE  1230    835         MOVAL   TOP_OF_KSTACK,SP        ; SP Points to the top of stack
                      00    5E  DA  1235    836         MTPR    SP,#PR$_KSP             ; Load the KSP with the top of stack
                                1238    837
                51    F3C4  CF  DE  1238    838         MOVAL   SCB,R1                  ; SET UP SYSTEM CONTROL
                                123D    839                                         ; BUFFER BASE ADDRESS
                      11    51  DA  123D    840         MTPR    R1,#PR$_SCBB            ; Load BASE System Control Block
                                1240    841                                         ; address
                      53    01  D0  1240    842         MOVL    #1,R3                   ; Initialize counter
                   50   82  AF  DE  1243    843         MOVAL   UNEX_E_I,R0             ; Load UNEXPECTED EXCEPTION or
                                1247    844                                         ; URGENT INTERRUPT address.
                      81    60  DE  1247    845 10$:    MOVAL   (R0),(R1)+              ; Load the SCB with trap catchers
             F5 53    0000007F  8F  F3  124A    846         AOBLEQ  #127,R3,10$             ; Continue until the SCB is full
                                1252    847
                      12    18  DA  1252    848         MTPR    #^X18,#PR$_IPL          ; Lower IPL to ^X18 (Urgent Interrupts)
                                1255    849
                            00  DD  1255    850         PUSHL   #0                      ; Push zero on PSL to change to
                                1257    851                                         ; KERNEL STACK
                0000125E'EF     DF  1257    852         PUSHAL  20$                     ; Push next Address on Stack
                            02  1255    853         REI                             ; Pop the Stack
                          A0'   11  125E    854 20$:    BRB     TEST_1                  ; Start test
                                1260    855
```

```
                        00000000   858          .PSECT  A_HARDCORE,BYTE,PIC
                             0000  859
                             0000  860          .SBTTL TEST 1: AOBLEQ, AOBLSS Test
                             0000  861 ;++
                             0000  862 ; Test description:
                             0000  863 ;
                             0000  864 ;       In this test, the AOBLEQ and AOBLSS instructions are functionally
                             0000  865 ;       verified.
                             0000  866 ;
                             0000  867 ; Assumptions:
                             0000  868 ;
                             0000  869 ;
                             0000  870 ;--
                             0000  871 START:
                             0000  872
                             0000  873 TEST_1:
                             0000  874 ;++
                             0000  875 ; Subtest 1: AOBLEQ
                             0000  876 ;--
                             0000  877 T1_S1:
    50   FFFFFFFF 8F    D0  0000  878          MOVL    #-1,R0                  ; Initialize counter
                   51    D4  0007  879          CLRL    R1                      ; Initialize index
                   50    D6  0009  880 10$:     INCL    R0                      ; Bump counter
         52    51   50    CD  000B  881          XORL3   R0,R1,R2                ; Counter = index?
                   06    13  000F  882          BEQL    20$                     ; Branch if so
          00000F29'EF    17  0011  883          JMP     ERROR_ROUTINE           ; Else, report the error
EA 51    0000007F 8F    F3  0017  884 20$:     AOBLEQ  #127,R1,10$             ; Do the AOBLEQ
   50    00000080 8F    D0  001F  885          MOVL    #128,R0                 ; Expect index = 128
         52    51   50    CD  0026  886          XORL3   R0,R1,R2                ; Is index = 128?
                   06    13  002A  887          BEQL    30$                     ; Branch if so
          00000F29'EF    17  002C  888          JMP     ERROR_ROUTINE           ; Else, report the error
                             0032  889 30$:                                    ; End of subtest
                             0032  890 ;++
                             0032  891 ; Subtest 2: AOBLSS
                             0032  892 ;--
                             0032  893 T1_S2:
    50   FFFFFFFF 8F    D0  0032  894          MOVL    #-1,R0                  ; Initialize counter
                   51    D4  0039  895          CLRL    R1                      ; Initialize index
                   50    D6  003B  896 10$:     INCL    R0                      ; Bump counter
         52    51   50    CD  003D  897          XORL3   R0,R1,R2                ; Counter = index?
                   06    13  0041  898          BEQL    20$                     ; Branch if so
          00000F29'EF    17  0043  899          JMP     ERROR_ROUTINE           ; Else, report the error
EA 51    00000080 8F    F2  0049  900 20$:     AOBLSS  #128,R1,10$             ; Do the AOBLSS
   50    00000080 8F    D0  0051  901          MOVL    #128,R0                 ; Expect index = 128
         52    51   50    CD  0058  902          XORL3   R0,R1,R2                ; Is index = 128?
                   06    13  005C  903          BEQL    30$                     ; Branch if so
          00000F29'EF    17  005E  904          JMP     ERROR_ROUTINE           ; Else, report the error
                             0064  905 30$:                                    ; End of subtest
                             0064  906 ;***** End of test *****
```

```
                                0064    908          .SBTTL TEST 2: SOBGEQ, SOBGTR Test
                                0064    909 ;++
                                0064    910 ; Test description:
                                0064    911 ;
                                0064    912 ;       In this test the instructions SOBGEQ and SOBGTR are functionally
                                0064    913 ;       verified.
                                0064    914 ;
                                0064    915 ; Assumptions:
                                0064    916 ;
                                0064    917 ;       Test 1
                                0064    918 ;--
                                0064    919 TEST_2:
                                0064    920 ;++
                                0064    921 ; Subtest 1: SOBGEQ
                                0064    922 ;--
                                0064    923 T2_S1:
 50   00000128 8F    D0          0064    924          MOVL    #^X128,R0          ; Initialize reference counter
 51   00000127 8F    D0          0068    925          MOVL    #^X127,R1          ; Initialize index operand
                50   D7          0072    926 10$:     DECL    R0                 ; Decrement reference counter
           52   51   50   CD     0074    927          XORL3   R0,R1,R2           ; Is R0 = R1?
                06   13          0078    928          BEQL    20$                ; Branch if so
          00000F29'EF   17       007A    929          JMP     ERROR_ROUTINE      ; Else, report the error
                EF   51   F4     0080    930 20$:     SOBGEQ  R1,10$             ; Should branch till R1 = -1
                50   D7          0083    931          DECL    R0                 ; Decrement reference counter
           52   51   50   CD     0085    932          XORL3   R0,R1,R2           ; Is R0 = R1?
                06   13          0089    933          BEQL    30$                ; Exit if so
          00000F29'EF   17       008B    934          JMP     ERROR_ROUTINE      ; Else, report the error
                                0091    935 30$:                                ; End of subtest
                                0091    936 ;++
                                0091    937 ; Subtest 2: SOBGTR
                                0091    938 ;--
                                0091    939 T2_S2:
 50   00000128 8F    D0          0091    940          MOVL    #^X128,R0          ; Initialize counter
 51   00000127 8F    D0          0098    941          MOVL    #^X127,R1          ; Initialize index operand
                50   D7          009F    942 10$:     DECL    R0                 ; Decrement counter
           52   51   50   CD     00A1    943          XORL3   R0,R1,R2           ; Is R1 = R0?
                06   13          00A5    944          BEQL    20$                ; Branch if so
          00000F29'EF   17       00A7    945          JMP     ERROR_ROUTINE      ; Else, report the error
                EF   51   F5     00AD    946 20$:     SOBGTR  R1,10$             ; Should branch until R1=0
                50   D7          00B0    947          DECL    R0                 ; Expect R1 = 0
           52   51   50   CD     00B2    948          XORL3   R0,R1,R2           ; Is R1 = 0?
                06   13          00B6    949          BEQL    30$                ; Exit if zero
          00000F29'EF   17       00B8    950          JMP     ERROR_ROUTINE      ; Else, report the error
                                00BE    951 30$:                                ; End of subtest
                                00BE    952 ;***** End of test *****
```

```
                                00BE    954          .SBTTL  TEST 3: BBS, BBC Test
                                00BE    955 ;++
                                00BE    956 ; Test descrption:
                                00BE    957 ;
                                00BE    958 ;       In this test, the instructions BBC and BBS are functionally verified.
                                00BE    959 ;
                                00BE    960 ; Assumptions:
                                00BE    961 ;
                                00BE    962 ;       Test 1-Test 2
                                00BE    963 ;--
                                00BE    964 TEST_3:
                                00BE    965 ;++
                                00BE    966 ; Subtest 1: BBS,BBC
                                00BE    967 ;--
                                00BE    968 T3_S1:
                          50    D4    00BE    969          CLRL    R0                      ; Initialize index
                          51    D4    00C0    970          CLRL    R1                      ; Initialize operand
                52  FFFFFFFF  8F    D0    00C2    971          MOVL    #-1,R2                  ; Initialize operand
                    04 51     50    E0    00C9    972 10$:     BBS     R0,R1,20$               ; Should not branch
                    06 51     50    E1    00CE    973          BBC     R0,R1,30$               ; Should branch
                  00000F29'EF       17    00D1    974 20$:     JMP     ERROR_ROUTINE           ; Else, report the error
                    04 52     50    E1    00D7    975 30$:     BBC     R0,R2,40$               ; Should not branch
                    06 52     50    E0    00DB    976          BBS     R0,R2,50$               ; Should branch
                  00000F29'EF       17    00DF    977 40$:     JMP     ERROR_ROUTINE           ; Else, report the error
                    E0 50     1F    F3    00E5    978 50$:     AOBLEQ  #31,R0,10$              ; Do for all 32 bit positions
                                      00E9    979                                          ; End of subtest
                                      00E9    980 ;***** End of test *****
```

```
                        00E9    982          .SBTTL  TEST 4: BBSS, BBCS Test
                        00E9    983 ;++
                        00E9    984 ; Test description:
                        00E9    985 ;
                        00E9    986 ;       In this test, the instructions BBSS and BBCS are functionally
                        00E9    987 ;       verified.
                        00E9    988 ;
                        00E9    989 ; Assumptions:
                        00E9    990 ;
                        00E9    991 ;       Test 1-Test 3
                        00E9    992 ;--
                        00E9    993 TEST_4:
                        00E9    994 ;++
                        00E9    995 ; Subtest 1: BBSS
                        00E9    996 ;--
                        00E9    997 T4_S1:
            53    D4    00E9    998          CLRL    R3                      ; Initialize position operand
            54    D4    00EB    999          CLRL    R4                      ; Initialize test operand
     04 54  53    E2    00ED   1000 10$:     BBSS    R3,R4,20$               ; Should not branch
     06 54  53    E2    00F1   1001          BBSS    R3,R4,30$               ; Should branch
00000F29'EF  17    00F5   1002 20$:     JMP     ERROR_ROUTINE           ; Else, report the error
     EE 55  1F    F3    00FB   1003 30$:     AOBLEQ  #31,R3,10$              ; Do until all 32 bits are set
                        00FF   1004                                          ; End of subtest
                        00FF   1005 ;++
                        00FF   1006 ; Subtest 2: BBCS
                        00FF   1007 ;--
                        00FF   1008 T4_S2:
            53    D4    00FF   1009          CLRL    R3                      ; Initialize position operand
            54    D4    0101   1010          CLRL    R4                      ; Initialize test operand
     06 54  53    E3    0103   1011 10$:     BBCS    R3,R4,20$               ; Should branch
00000F29'EF  17    0107   1012          JMP     ERROR_ROUTINE           ; Else, report the error
     03 54  53    E3    010D   1013 20$:     BBCS    R3,R4,30$               ; Should not branch
          0006   31    0111   1014          BRW     40$                     ; Branch if okay
00000F29'EF  17    0114   1015 30$:     JMP     ERROR_ROUTINE           ; Else, report the error
     E5 53  1F    F3    011A   1016 40$:     AOBLEQ  #31,R3,10$              ; Do until all 32 are set
                        011E   1017                                          ; End of subtest
                        011E   1018 ;***** End of test *****
```

```
                                011E  1020          .SBTTL  TEST 5: BLBS, BLBC Test
                                011E  1021 ;++
                                011E  1022 ; Test description:
                                011E  1023 ;
                                011E  1024 ;       In this test the instructions BLBS and BLBC are functionally verified.
                                011E  1025 ;
                                011E  1026 ; Assumptions:
                                011E  1027 ;
                                011E  1028 ;       Test 1-Test 5
                                011E  1029 ;--
                                011E  1030 TEST_5:
                                011E  1031 ;++
                                011E  1032 ; Subtest 1: BLBS, BLBC
                                011E  1033 ;--
                                011E  1034 T5_S1:
                         50  D4 011E  1035          CLRL    R0                      ; Initialize test operand
                         51  D4 0120  1036          CLRL    R1                      ; Initialize test operand
                      52  01  D0 0122  1037          MOVL    #1,R2                   ; Initialize test operand
                53 FFFFFFFE  8F  D0 0125  1038          MOVL    #-2,R3                  ; Initialize test operand
                         03 50  E8 012C  1039          BLBS    R0,10$                  ; Should not branch
                         06 50  E9 012F  1040          BLBC    R0,20$                  ; Should branch
                00000F29'EF  17 0132  1041 10$:    JMP     ERROR_ROUTINE           ; Else, report the error
                         03 52  E9 0138  1042 20$:    BLBC    R2,30$                  ; Should not branch
                         06 52  E8 013B  1043          BLBS    R2,40$                  ; Should branch
                00000F29'EF  17 013E  1044 30$:    JMP     ERROR_ROUTINE           ; Else, report the error
                         03 53  E8 0144  1045 40$:    BLBS    R3,50$                  ; Should not branch
                         06 53  E9 0147  1046          BLBC    R3,60$                  ; Should branch
                00000F29'EF  17 014A  1047 50$:    JMP     ERROR_ROUTINE           ; Else, report the error
                                0150  1048 60$:                                    ; End of subtest
                                0150  1049 ;***** End of test *****
```

```
                                0150  1051          .SBTTL  Diagnostic Subroutines
                                0150  1052 ;++
                                0150  1053 ; Endpass
                                0150  1054 ;
                                0150  1055 ; Here is the endpass routine.
                                0150  1056 ;
                                0150  1057 ;--
                                0150  1058 END_PASS:
                00000174'EF   16  0150  1059          JSB     CLEAN_UP                ; CLEANUP THE CPU
                00000213'EF   D6  0156  1060          INCL    PASS_COUNT              ; Bump the pass count once
50  00000213'EF 00000064 8F   7B  015C  1061          EDIV    #100,PASS_COUNT,R0,R1   ; GET NUMBER OF 100 PASSES
                        51        0168
                        51    D5  0169  1062 10$:     TSTL    R1                      ; DONE 100 PASSES?
                        03    12  016B  1063          BNEQ    20$                     ; BRANCH IF NO
                    FE90'     31  016D  1064          BRW     TEST_6
                FE8C CF       17  0170  1065 20$:     JMP     START
                                0174  1066
                                0174  1067 ;++
                                0174  1068 ; Cleanup
                                0174  1069 ;--
                                0174  1070 CLEAN_UP:
                12    1F      DA  0174  1071          MTPR    #31,#IPL                ; SET IPL TO 31
                                05  0177  1072          RSB                             ; RETURN
                                0178  1073
                                0178  1074          .END
```

```
BAD_DISK                        ********    X   01
BOO$C_COMBUFSZ                = 00000084
BOO$GL_FREEMEM                  0000055E  R     01
BOO$GT_COMBUF                   000004DA  R     01
BOO$READPROMPT                  00000E00  RG    01
BOO$TYPE_ASCIC                  00000EB7  R     01
BUF                           = 0000000C
CLEAN_UP                        00000174  R     03
CONTROL_U                     = 00000015
CPU_BAD                         0000020F  R     01
CPU_ERR                         00000F93  R     01
CR                            = 0000000D
DISK_ERR                        00000F88  R     01
DRIVE_NFND                      00000F74  R     01
DS$GB_UDA50_INIT                ********    X   01
END_PASS                        00000150  R     03
ERROR_ROUTINE                   00000F29  RG    01
ERR_END                         00000F9C  R     01
HMBLK_ERR                       ********    X   01
IDC_CSR                       = 00F26200    G
INIT1                           00001220  R     01
INIT_DRV0                       ********    X   01
INTERRUPT_STACK                 00000A00  RG    01
IPL                           = 00000012
KERNEL_STACK                    00000800  RG    01
LESI_UDA_PARAM                  ********    X   01
LF                            = 0000000A
LOAD_BAD                        00000F7E  R     01
LOAD_ERROR                      0000020B  R     01
MCHK_IDC                        000011F4  RG    01
MCHK_LES_UDA                    00001208  RG    01
MCKADD                        = 00000004
NOCNTLR_FND                     00000207  RG    01
NODRIVE_0_1                     ********    X   01
NO_LOAD                         00000F6A  RG    01
OUTBSLSH                        00000E7E  R     01
OUTCHAR                         00000E87  R     01
OUTEX                           00000E91  R     01
OUTR8                           00000E84  R     01
OUTZSTR                         00000E92  R     01
PASS_COUNT                      00000213  R     01
PR$_IPL                       = 00000012
PR$_ISP                       = 00000004
PR$_KSP                       = 00000000
PR$_MAPEN                     = 00000038
PR$_MCESR                     = 00000026
PR$_RXCS                      = 00000020
PR$_RXDB                      = 00000021
PR$_SCBB                      = 00000011
PR$_TXCS                      = 00000022
PR$_TXDB                      = 00000023
PRIM_REV                      = 00000001
PRINTABORT                      00000257  R     01
PRINTAUTO                       0000045F  R     01
PRINTCPU                        0000049F  R     01
PRINTDAA0                       000004B0  R     01
PRINTDISKERR                    0000030F  R     01
PRINTDJA0                       000004BC  R     01
PRINTDJA1                       000004C2  R     01
PRINTDRVERR                     000002D0  R     01
PRINTDUA0                       00000486  R     01
PRINTEND                        00000267  R     01
PRINTFAIL                       00000245  R     01
PRINTFIELD                      00000232  R     01
PRINTLOADMCK                    0000031C  R     01
PRINTMENU                       00000465  R     01
PRINTNOCNTLR                    00000351  R     01
PRINTNODRIVE01                  00000376  R     01
PRINTNSPN6025                   000003DB  R     01
PRINTNSPN8081                   00000427  R     01
PRINTPART                       C000046B  R     01
PRINTPASS                       00000223  R     01
PRINTPROPER                     000003AD  R     01
PRINTRA60                       00C004C8  R     01
PRINTRA80                       000004CE  R     01
PRINTRA81                       000004D4  R     01
PRINTRC25                       000004AA  R     01
PRINTRET                        00000293  R     01
PRINTSPNEND                     00000456  R     01
PRINTTEND                       00000285  R     01
PRINTTEST                       00000475  R     01
PRINT_ROUTINE                   00000F0E  R     01
PROMPT                        = 00000004
PROMPT1                         00000E9B  R     01
PROMPT_MESS                     00000ED3  R     01
RA60_MESS_ST                    00000217  R     01
RA60_TEST                       00000FE0  RG    01
RA8081_MESS_ST                  0000021B  R     01
RA8081_TEST                     0000109E  RG    01
RA81_PRES                       ********    X   01
RC25_MESS_ST                    0000021F  R     01
RC25_TEST                       00001121  RG    01
RPB_BUFF                        00000C00  RG    01
RUBOUT                        = 0000007F
SCB                             00000600  RG    01
SCB0                            00000600  RG    01
SCB20                           00000620  RG    01
SCB24                           00000624  RG    01
SCB4                            00000604  RG    01
SCB60                           00000660  RG    01
SCB64                           00000664  RG    01
SCB8                            00000608  RG    01
SEC_REV                       = 00000003
SIZE                          = 00C00008
SPIN_DWN                        ********    X   01
SS$_NORMAL                    = 00000001
START                           00000000  R     03
T1_S1                           00000000  R     03
T1_S2                           00000032  R     03
T2_S1                           00000064  R     03
T2_S2                           00000091  R     03
T3_S1                           000000BE  R     03
T4_S1                           000000E9  R     03
T4_S2                           000000FF  R     03
```

```
T5_S1                           0000011E R      03
TEST_1                          00000000 R      03
TEST_2                          00000064 R      03
TEST_3                          000000BE R      03
TEST_4                          000000E9 R      03
TEST_5                          0000011E R      03
TEST_6                          ********   X    03
TEST_8                          ********   X    01
TOP_OF_ISTACK                   00000BFC RG     01
TOP_OF_KSTACK                   000009FC RG     01
TXCS                          = 00000022
TXDB                          = 00000023
UDA_LESI                        ********   X    01
UDA_LESI_CSR                  = 40FFF468   G
UD_DRIVER1                      ********   X    01
UNEX_E_I                        000011C8 RG     01
V_RUB                         = 00000000
WHAT_DRIVE                      ********   X    01
WHAT_MODE                       00000203 RG     01
```

Psect synopsis

PSECT name	Allocation		PSECT No.		Attributes									
. ABS .	00000000	(0.)	00	(0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC BYTE
A_ODT	00001260	(4704.)	01	(1.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC PAGE
A_BASE	00000000	(0.)	02	(2.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC BYTE
A_HARDCORE	00000178	(376.)	03	(3.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC BYTE

! Symbol Cross Reference !

SYMBOL	VALUE	DEFINITION	REFERENCES...
BAD_DISK	00000000-XR		#-548 (4) #-629 (4) #-688 (4) #-724 (4)
BOO$C_COMBUFSZ	=00000084	219 (4)	#-449 (4)
BOO$GL_FREEMEM	0000055E-R	259 (4)	#-476 (4)
BOO$GT_COMBUF	000004DA-R	258 (4)	448 (4) 491 (4)
BOO$READPROMPT	00000E00-R	370 (4)	451 (4) 481 (4)
BOO$TYPE_ASCIC	00000EB7-R	473 (4)	#-492 (4)
BUF	=0000000C	360 (4)	#-377 (4)
CLEAN_UP	00000174-R	1070 (12)	1059 (12) 598 (4)
CONTROL_U	=00000015	327 (4)	#-402 (4)
CPU_BAD	0000020F-R	223 (4)	#-763 (4)
CPU_ERR	00000F93-R	572 (4)	#-550 (4)
CR	=0000000D	325 (4)	#-381 (4) #-409 (4)
DISK_ERR	00000F88-R	567 (4)	#-548 (4)
DRIVE_NEND	00000F74-R	556 (4)	#-544 (4)
DS$GB_UDA50_INIT	00000000-XR		#-493 (4)
END_PASS	00000150-R	1058 (12)	
ERROR_ROUTINE	00000F29-R	528 (4)	1002 (10) 1012 (10) 1015 (10) 1041 (11) 1044 (11) 1047 (11) 630 (4) 689 (4) 725 (4) 764 (4) 767 (4) 796 (4) 883 (7) 888 (7) 899 (7) 904 (7) 929 (8) 934 (8) 945 (8) 950 (8) 974 (9) 977 (9)
ERR_END	00000F9C-R	576 (4)	#-554 (4) #-560 (4) #-566 (4) #-571 (4)
HMBLK_ERR	00000000-XR		#-647 (4) #-736 (4)
IDC_CSR	=00F26200	217 (4)	
INIT1	00001220-R	827 (6)	#-204 (4)
INIT_DRV0	00000000-XR		#-668 (4) #-710 (4)
INTERRUPT_STACK	00000A00-R	281 (4)	
IPL	=00000012	211 (4)	#-1071 (12)
KERNEL_STACK	00000800-R	276 (4)	
LES_UDA_PARAM	00000000-XR		#-692 (4)
LF	=0000000A	326 (4)	#-411 (4)
LOAD_BAD	00000F7E-R	562 (4)	#-546 (4)
LOAD_ERROR	00000208-R	222 (4)	#-546 (4) #-766 (4)
MCHK_IDC	000011F4-R	778 (4)	
MCHK_LES_UDA	00001208-R	792 (4)	
MCKADD	=00000004	212 (4)	
NOCNTLR_END	00000207-R	221 (4)	#-542 (4) #-795 (4)
NODRIVE_0_1	00000000-XR		#-544 (4)
NO_LOAD	00000F6A-R	551 (4)	#-542 (4)
OUTBSLSH	00000E7E-R	418 (4)	#-391 (4) #-401 (4)
OUTCHAR	00000E87-R	422 (4)	#-382 (4) #-412 (4) #-420 (4) #-424 (4) #-430 (4)
OUTEX	00000E91-R	426 (4)	#-429 (4)
OUTRB	00000E84-R	421 (4)	#-397 (4) #-407 (4)
OUTZSTR	00000E92-R	427 (4)	#-373 (4) #-431 (4)
PASS_COUNT	00000213-R	224 (4)	#-1060 (12) #-1061 (12) #-761 (4)
PR$_IPL	=00000012		#-848 (6)
PR$_ISP	=00000004		#-833 (6)
PR$_KSP	=00000000		#-836 (6)

```
PR$_MAPEN                =00000038                      #-827    (6)
PR$_MCESR                =00000026                      #-759    (4)    #-779    (4)    #-793    (4)
PR$_RXCS                 =00000020                      #-384    (4)
PR$_RXDB                 =00000021                      #-386    (4)
PR$_SCBB                 =00000011                      #-840    (6)
PR$_TXCS                 =00000022                      #-423    (4)
PR$_TXDB                 =00000023                      #-425    (4)
PRIM_REV                 =00000001      65    (3)
PRINTABORT                00000257-R    231   (4)      539    (4)
PRINTAUTO                 0000045F-R    244   (4)      533    (4)      585    (4)
PRINTCPU                  0000049F-R    248   (4)      573    (4)
PRINTDAAO                 000004B0-R    250   (4)      720    (4)      732    (4)
PRINTDISKERR              0000030F-R    236   (4)      568    (4)
PRINTDJAO                 000004BC-R    252   (4)      625    (4)      643    (4)
PRINTDJA1                 000004C2-R    253   (4)      622    (4)      640    (4)
PRINTDRVERR               000002D0-R    235   (4)      648    (4)      737    (4)
PRINTDUAO                 000004B6-R    251   (4)      684    (4)
PRINTEND                  00000267-R    232   (4)      580    (4)
PRINTFAIL                 00000245-R    230   (4)      531    (4)
PRINTFIELD                00000232-R    229   (4)      577    (4)      650    (4)      739    (4)
PRINTLOADMCK              0000031C-R    237   (4)      563    (4)
PRINTMENU                 00000465-R    245   (4)      536    (4)      589    (4)
PRINTNOCNTLR              00000351-R    238   (4)      552    (4)
PRINTNODRIVE01            00000376-R    239   (4)      557    (4)
PRINTNSPN6025             000003DB-R    241   (4)      635    (4)      730    (4)
PRINTNSPN8081             00000427-R    242   (4)      695    (4)
PRINTPART                 0000046B-R    246   (4)      619    (4)      682    (4)      718    (4)
PRINTPASS                 00000223-R    228   (4)      614    (4)      671    (4)      713    (4)
PRINTPROPER               000003AD-R    240   (4)      633    (4)      693    (4)      728    (4)
PRINTRA60                 000004C8-R    254   (4)      617    (4)
PRINTRA80                 000004CE-R    255   (4)      680    (4)
PRINTRA81                 000004D4-R    256   (4)      676    (4)
PRINTRC25                 000004AA-R    249   (4)      716    (4)
PRINTRET                  00000293-R    234   (4)      595    (4)
PRINTSPNEND               00000456-R    243   (4)      645    (4)      652    (4)      734    (4)      741    (4)
PRINTTEND                 00000285-R    233   (4)      592    (4)
PRINTTEST                 00000475-R    247   (4)      627    (4)      686    (4)      722    (4)
PRINT_ROUTINE             00000F0E-R    505   (4)      532    (4)      534    (4)      537    (4)      540    (4)
                                                       553    (4)      559    (4)      565    (4)      570    (4)
                                                       574    (4)      578    (4)      581    (4)      586    (4)
                                                       590    (4)      593    (4)      596    (4)      615    (4)
                                                       618    (4)      620    (4)      623    (4)      626    (4)
                                                       628    (4)      634    (4)      636    (4)      641    (4)
                                                       644    (4)      646    (4)      649    (4)      651    (4)
                                                       653    (4)      672    (4)      677    (4)      681    (4)
                                                       683    (4)      685    (4)      687    (4)      694    (4)
                                                       696    (4)      714    (4)      717    (4)      719    (4)
                                                       721    (4)      723    (4)      729    (4)      731    (4)
                                                       733    (4)      735    (4)      738    (4)      740    (4)
                                                       742    (4)
PROMPT                   =00000004      352   (4)    #-372    (4)
PROMPT1                   00000E98-R    446   (4)    #-489    (4)
PROMPT_MESS               00000ED3-R    488   (4)      654    (4)      697    (4)      743    (4)
RA60_MESS_ST              00000217-R    225   (4)    #-613    (4)    #-616    (4)
RA60_TEST                 00000FE0-R    611   (4)
RA80RT_MESS_ST            0000021B-R    226   (4)    #-670    (4)    #-673    (4)
RA80RT_TEST               0000109E-R    666   (4)
```

```
                                CRD HARDCORE INSTRUCTION TESTS
R48?_PRES                       C0000000-XR                     #-674     (4)
RC??_MESS_ST                    0000021F-R      227     (4)     #-712     (4)     #-715     (4)
RC??_TEST                       00001121-R      708     (4)
RPB_BUFF                        00000C00-R      286     (4)
RUNOUT                          =0000007F       328     (4)     #-388     (4)
SCB                             00000600-R      264     (4)       838     (6)
SCBO                            00000600-R      265     (4)
SCB??0                          00000620-R      268     (4)
SCB??4                          00000624-R      269     (4)
SCB?                            00000604-R      266     (4)
SCB??0                          00000660-R      270     (4)
SCB??4                          00000664-R      271     (4)
SCB?                            00000608-R      267     (4)
SEC_PEV                         =00000003       66      (3)
SIZE                            =00000008       355     (4)     #-374     (4)     #-395     (4)
SPIN_DWN                        00000000-XR                     #-631     (4)     #-690     (4)     #-726     (4)
SSS_NORMAL                      =00000001                       #-416     (4)
STAR                            00000000-R      871     (7)       1065    (12)
T1_S                            00000000-R      877     (7)
T1_S                            00000032-R      893     (7)
T2_S                            00000064-R      923     (8)
T2_S                            00000091-R      939     (8)
T3_S                            000000BE-R      968     (9)
T4_S                            000000E9-R      997     (10)
T4_S                            000000FF-R      1008    (10)
T5_S                            0000011E-R      1034    (11)
TEST                            00000000-R      873     (7)     #-854     (6)
TEST                            00000064-R      919     (8)
TEST                            000000BE-R      964     (9)
TEST                            000000E9-R      993     (10)
TEST                            0000011E-R      1030    (11)
TEST                            00000000-XR                     #-1064    (12)
TEST                            00000000-XR                     #-632     (4)     #-727     (4)
TOP_OF_ISTACK                   00000BFC-R      282     (4)       832     (6)
TOP_OF_KSTACK                   000009FC-R      277     (4)       835     (6)
TXCS                            =00000022       213     (4)     #-510     (4)
TXDB                            =00000023       214     (4)     #-512     (4)
UDA_??SI                        00000000-XR                       781     (4)
UDA_??_CSR                      =40FFF468       218     (4)
UD_DRIVER1                      00000000-XR                       494     (4)
UNEX_I                          000011C8-R      758     (4)       843     (6)
V_RUN                           =00000000       329     (4)     #-390     (4)     #-400     (4)
WHAT_DRIVE                      00000000-XR                     #-621     (4)     #-638     (4)     #-667     (4)     #-709     (4)
WHAT_MODE                       00000203-R      220     (4)     #-529     (4)     #-583     (4)     #-612     (4)     #-669     (4)
                                                                #-711     (4)     #-829     (6)
```

Macros Cross Reference

MACRO	SIZE	DEFINITION	REFERENCES...
$DEFIN1	1	313 (4)	313 (4) 314 (4) 315 (4)
$PP730DEF	1	314 (4)	314 (4)
$PRDEF	4	313 (4)	313 (4)
$SSDEF	21	315 (4)	315 (4)
ABORT_MESSAGE	1	89 (4)	231 (4)
AUTO_MESSAGE	1	152 (4)	244 (4)
CPU_MESSAGE	1	93 (4)	248 (4)
DAAD_MESSAGE	1	184 (4)	250 (4)
DISKERR_MESSAGE	1	113 (4)	236 (4)
DJAC_MESSAGE	1	188 (4)	252 (4)
DJAI_MESSAGE	1	192 (4)	253 (4)
DRVERR_MESSAGE	1	139 (4)	235 (4)
DUAC_MESSAGE	1	196 (4)	251 (4)
END_MESSAGE	1	117 (4)	232 (4)
FAIL_MESSAGE	1	85 (4)	230 (4)
FIELD_MESSAGE	1	97 (4)	229 (4)
LOADMCK_MESSAGE	1	109 (4)	237 (4)
MENU_MESSAGE	1	156 (4)	245 (4)
NOCNTLR_MESSAGE	1	101 (4)	238 (4)
NODRIVE01_MESSAGE	1	105 (4)	239 (4)
NSPN6025_MESSAGE	1	134 (4)	241 (4)
NSPN8081_MESSAGE	1	148 (4)	242 (4)
PART_MESSAGE	1	164 (4)	246 (4)
PASS_MESSAGE	1	81 (4)	228 (4)
PROBER_MESSAGE	1	130 (4)	240 (4)
RA60_MESSAGE	1	172 (4)	254 (4)
RA80_MESSAGE	1	176 (4)	255 (4)
RA81_MESSAGE	1	180 (4)	256 (4)
RC25_MESSAGE	1	168 (4)	249 (4)
RET_MESSAGE	1	125 (4)	234 (4)
SPNEND_MESSAGE	1	144 (4)	243 (4)
TEND_MESSAGE	1	121 (4)	233 (4)
TEST_MESSAGE	1	160 (4)	247 (4)

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	112	00:00:00.44	00:00:04.21
Command processing	110	00:00:00.40	00:00:02.94
Pass 1	814	00:00:10.00	00:01:14.44
Symbol table sort	6	00:00:01.08	00:00:12.34
Pass 2	499	00:00:03.33	00:00:32.67
Symbol table output	18	00:00:00.13	00:00:01.27
Psect synopsis output	6	00:00:00.03	00:00:00.03
Cross-reference output	132	00:00:00.90	00:00:08.37
Assembler run totals	1701	00:00:16.33	00:02:16.45

The working set limit was 500 pages.
72173 bytes (141 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 620 non-local and 69 local symbols.
1074 source lines were read in Pass 1, producing 21 object records in Pass 2.
48 pages of virtual memory were used to define 38 macros.

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

Macro library name	Macros defined
SYS$SYSROOT:[SYSLIB]STARLET.MLB;1	6

568 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

MACRO/LIST/CROSS/OBJECT ENKAZ01A

Table of contents

```
0000     1          .TITLE  ENKAZ02 - UDA50, LESI, IDC SIZING TESTS
0000     2          .IDENT  '01-30'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1983 BY                                  *
0000     8  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9  ;*  ALL RIGHTS RESERVED.                                                     *
0000    10  ;*                                                                           *
0000    11  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    12  ;*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE            *
0000    13  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER       *
0000    14  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    15  ;*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY         *
0000    16  ;*  TRANSFERRED.                                                              *
0000    17  ;*                                                                           *
0000    18  ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE      *
0000    19  ;*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
0000    20  ;*  CORPORATION.                                                              *
0000    21  ;*                                                                           *
0000    22  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS      *
0000    23  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24  ;*                                                                           *
0000    25  ;*                                                                           *
0000    26  ;*****************************************************************************
0000    27  ;
0000    28
0000    29  ;++
0000    30  ; FACILITY:     BOOTS
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;       This module contains the bootstrap device driver for the
0000    34  ;       UDA 50  and RCF25/RC25 disks. It will perform some sizing
0000    35  ;       for supported CRD 11/730 disk options and read the label
0000    36  ;       off RA60 and RC25 disks.
0000    37  ;
0000    38  ; ENVIRONMENT:  IPL 31, kernel mode, code must be PIC
0000    39  ;
0000    40  ; AUTHOR:  Kerbey T. Altmann,   CREATION DATE:  20-Nov-1981
0000    41  ;
0000    42  ; MODIFIED BY:
0000    43  ;
0000    44  ;       Peter Green     1-April-1983
0000    45  ;       Modified to run with ENKAZ level 4 program in CRD package.
0000    46  ;
0000    47  ;       Darryl L. Ball  14-June-1984
0000    48  ;
0000    49  ;       Added 730/725 Processor register definitions. Also modified existing
0000    50  ;       boot routine to Boot the system disk if no RL02 is present.
0000    51  ;
0000    52  ;--
```

```
                        0000     54          .SBTTL  DECLARATIONS
                        0000     55 ;
                        0000     56 ; INCLUDE FILES:
                        0000     57 ;
                        0000     58
                        0000     59          $BTDDEF                         ; Boot device types
                        0000     60          $IODEF                          ; I/O function codes
                        0000     61          $MSCPDEF                        ; MSCP definitions
                        0000     62          $PRDEF                          ; Processor registers
                        0000     63          $PR730DEF                       ; 730/725 Processor registers
                        0000     64          $PTEDEF                         ; Page table entries
                        0000     65          $RPBDEF                         ; RPB offsets
                        0000     66          $SSDEF                          ; Status codes
                        0000     67          $UBADEF                         ; UBA definitions
                        0000     68          $UBIDEF                         ; 11/750 UBA definitions
                        0000     69          $VADEF                          ; Virtual addresses
                        0000     70
                        0000     71 ;
                        0000     72 ; EQUATED SYMBOLS:
                        0000     73 ;
00000000                0000     74          UDAIP   = 0
00000002                0000     75          UDASA   = 2
00000001                0000     76          GO      = 1
00008000                0000     77          OWN     = 1@15
0000000B                0000     78          S1      = 11
0000000E                0000     79          S4      = 14
000001EE                0000     80          UMR     = 494               ; Last-1 UNIBUS Mapping Register
                        0000     81
                        0000     82 ;
                        0000     83 ; OWN STORAGE:
                        0000     84 ;
                        0000     85
                        0000     86 ;
                        0000     87 ; Boot driver table entry
                        0000     88 ;
                        0000     89
                    00000000     90 .Psect  SIZING ,         page    ; Define psect allocation
                        0000     91
                        0000     92          .SBTTL  UDA50 Bootstrap device initialization
                        0000     93
                        0000     94 ;++
                        0000     95 ;
                        0000     96 ; Inputs:
                        0000     97 ;
                        0000     98 ;        R9 -->  RPB
                        0000     99 ;
                        0000    100 ; Outputs:
                        0000    101 ;
                        0000    102 ;        R0 - status code
                        0000    103 ;
                        0000    104 ;--
                        0000    105
                    01FC0000    106 .Entry UD_INIT, ^M<R2,R3,R4,R5,R6,R7,R8>
                        0002    107
                        0002    108          .ENABLE LSB
                        0002    109
50   38   DB            0002    110          MFPR    #PR$_MAPEN, R0          ; Get the mapping status
```

```
                                0005   111 ;
                                0005   112 ; Set up a UNIBUS mapping register(s) to cover the ring and buffers.
                                0005   113 ; To make things easy, we will grab the last two register (494 & 495).
                                0005   114 ; These registers are necessary since the ring area will be accessed by
                                0005   115 ; the controller which is a UNIBUS device that does not do any mapping.
                                0005   116 ;
 56   0003DC00 8F    D0  0005   117          MOVL    #<UMR@9>, R6                    ; Set up a constant
      52   0200'CF    9E  000C   118          MOVAB   W^INTTBL, R2                    ; Get the address of the ring
  51   52    77'AF    CB  0011   119          BICL3   B^BYTE_OFF, R2, R1              ; Get the byte offset in page
  02 A2   51    5o    C9  0016   120          BISL3   R6, R1, 2(R2)                   ; Set in the UNIBUS addr
     02 A2   10'    C0  001B   121          ADDL    S^#<RING-INTTBL>,2(R2)          ; Make it the ring address
52   52    15    09    EF  001F   122          EXTZV   #VA$V_VPN,#VA$S_VPN,R2,R2 ; Get the page frame
                                0024   123          ASSUME  RPB$L_ADPVIR EQ RPB$L_ADPPHY+4
      53   5C A940    D0  0024   124          MOVL    RPB$L_ADPPHY(R9)[R0],R3 ; Get correct pointer to UBA reg
                                0029   125          ASSUME  RPB$L_CSRVIR EQ RPB$L_CSRPHY+4
      57   54 A940    D0  0029   126          MOVL    RPB$L_CSRPHY(R9)[R0],R7 ; Get correct address of device CSR
           0C 50      E9  002E   127          BLBC    R0,20$                          ; If clr, then physical
      52   50 B942    D0  0031   128          MOVL    @RPB$L_SVASPT(R9)[R2],R2; Virtual, get physical
52   FFE00000 8F    CA  0036   129          BICL    #^C<PTE$M_PFN>,R2               ; Now a physical PFN
      54   0FB8 C3    DE  003D   130 20$:     MOVAL   UBA$L_MAP+<UMR*4>(R3),R4; Get the last two UMR's
 84   52    AC'AF    C9  0042   131          BISL3   B^VALID,R2,(R4)+                ; Set as valid w/PFN
           52         D6  0047   132          INCL    R2                              ; Set next page just in case
 64   52    AC'AF    C9  0049   133          BISL3   B^VALID,R2,(R4)                 ; Set as valid w/PFN
                                004E   134 ;
                                004E   135 ; Now go thru the ridiculously complicated startup sequence.  This is a
                                004E   136 ; fugue in four parts.
                                004E   137 ;
      58   02    D0  004E   138          MOVL    #2,R8                           ; Make two tries at this
53    0200'CF    9E  0051   139 RETRY:  MOVAB   W^INTTBL, R3
      50   0B    D0  0056   140          MOVL    #S1, R0                         ; Step flag
           67    B4  0059   141          CLRW    UDAIP(R7)                       ; Poke the controller's CSR
                            0058   142 ;
                            0058   143 ; Wait 100 microseconds.
                            0058   144 ;
      52   1B    DB  0058   145 TIME:    MFPR    #PR730$_TODR, R2                ; Current time
52    03E8 C2    9E  005E   146          MOVAB   1000(R2), R2                    ; Set for 10 seconds later
  54    02 A7    B0  0063   147 LOOP:    MOVW    UDASA(R7),R4                    ; Check the status register
        39       19  0067   148          BLSS    ERROR                           ; Bit 15 set is the error indicator
0E 54   50       E0  0069   149          BBS     R0,R4,30$                       ; Done with this step?
      51   1B    DB  006D   150          MFPR    #PR730$_TODR, R1                ; No, pick up time
      52   51    D1  0070   151          CMPL    R1,R2                           ; Are we past due time?
           2D    1A  0073   152          BGTRU   ERROR                           ; Yep, error
           EC    11  0075   153          BRB     LOOP                            ; No, try again
                            0077   154
                            0077   155 BYTE_OFF:
           FFFFFE00         0077   156          .LONG   ^C<^X1FF>                       ; Mask for byte offset in page
                            007B   157
   02 A7   83    B0  007B   158 30$:     MOVW    (R3)+,UDASA(R7)                 ; Send the controller the next step
   D8 50   0E    F3  007F   159          AOBLEQ  #S4,R0,TIME                     ; Set for next step
                            0083   160 ;
                            0083   161 ; Initialization complete.  Write the packet address in the ring.
                            0083   162 ;
                            0083   163
      51   0250'CF    9E  0083   164          MOVAB   W^RSPPKT, R1                    ; Get the address of response packet
        51   EC AF    CA  0088   165          BICL    BYTE_OFF, R1
0210'CF    51   56    C9  008C   166          BISL3   R6, R1, W^RD                    ; Set byte offset in ring desc
      51   021C'CF    9E  0092   167          MOVAB   W^CMDPKT, R1                    ; Get the address of command packet
```

```
                51   DD AF   CA  0097   168          BICL    BYTE_OFF, R1
          0214'CF    51   56   C9  009B   169          BISL3   R6, R1,W^CD             ; Set byte offset in ring desc
                                04  00A1   170          RET
                                    00A2   171
                       AC 58    F5  00A2   172 ERROR:   SOBGTR  R8,RETRY               ; Try once again
                50   0054 8F    3C  00A5   173          MOVZWL  #SS$_CTRLERR,R0         ; Set failure status
                                04  00AA   174          RET
                                    00AB   175
                                    00AB   176          .DISABLE LSB
                          000000AC  00AB   177          .=<.+1>&-2
                          80000000  00AC   178 VALID:   .LONG   ^X80000000            ; SIgn bit set
                                    00B0   179
                                    00B0   180 Ds$Gb_Uda50_Init::                   ; Global label
                                    00B0   181 Init:                                ; convenient local label
                                00  00B0   182          .Byte   0                   ; .. init flag
                                    00B1   183 ;
                                    00B1   184 ; RINGS
                                    00B1   185 ;
                                    00B1   186 ;
                                    00B1   187
                                    00B1   188 .Align  Page
                                    0200   189
                              8000  0200   190 INTTBL: .WORD   OWN                   ; Step 1 pattern
                          0000C0000 0202   191          .LONG   0                   ; Step 2 & 3 pattern
                              0001  0206   192          .WORD   GO
                                    0208   193
                         0000 0000  0208   194          .WORD   0,0                 ; Reserved
                              0000  020C   195 CMDINT:  .WORD   0                   ; Command status word
                              0000  020E   196 RSPINT:  .WORD   0                   ; Response status word
                                    0210   197 RING:
                         00000000   0210   198 RD:      .LONG   0                   ; UNIBUS address of response ring
                         00000000   0214   199 CD:      .LONG   0                   ; UNIBUS address of command ring
                                    0218   200 ;
                              0030  0218   201          .WORD   48                  ; Length of message
                              0001  021A   202          .WORD   1                   ; ID
                              0001  021C   203 CMDPKT:  .WORD   1
                         0000024C   021E   204          .BLKW   23                  ; Full envelope
                                    024C   205 ;
                         00000250   024C   206          .BLKW   2
                                    0250   207 UDA_Response::                       ; Global label or CONFIG
                         00000280   0250   208 RSPPKT:  .BLKW   24
```

```
                                        .SBTTL  UDA50 Bootstrap driver Q10
                                ;+
                                ; Inputs:
                                ;
                                ;       R3      - base address of adapter's register space
                                ;       R5      - lbn for current piece of transfer
                                ;       R6      - contains 0
                                ;       R7      - address of the device's CSR
                                ;       R8      - size of transfer in bytes
                                ;       R9      - address of the RPB
                                ;       R10     - starting address of transfer (byte offset in first
                                ;                 page ORed with starting map register number)
                                ;       R11     - LBN at start of transfer
                                ;
                                ;       FUNC(AP)- I/O operation (IO$_READLBLK)
                                ;       SIZE(AP)- Size of transfer in bytes
                                ;       MODE(AP)- Address interpretation mode (0 = physical, 1 = virtual)
                                ;
                                ; Implicit inputs:
                                ;
                                ;       RPB$W_UNIT      - RPB field containing boot device unit number
                                ;
                                ; Outputs:
                                ;
                                ;       R0 - status code
                                ;
                                ;               SS$_NORMAL      - successful transfer
                                ;               SS$_NOSUCHDEV   - unsupported device
                                ;               SS$_CTRLERR     - fatal controller error
                                ;
                                ;       R3 - must be preserved
                                ;
                                ;
                                ;--

00000010                        FUNC = 16
00000014                        MODE = 20

50445243                        CRDPACK1 = ^X50445243   ; CRDP
204B4341                        CRDPACK2 = ^X204B4341   ; ACK_

00000000                WHAT_DRIVE::    .LONG 0
      00                RA81_PRES::     .BYTE 0
      00                RC25_PRES::     .BYTE 0
      00                NODRIVE_0_1::   .BYTE 0
      00                HMBLK_ERR::     .BYTE 0
      00                BAD_DISK::      .BYTE 0
      00                SPIN_DWN::      .BYTE 0
                        ;
                        ;
                        ;

00000081                C_D_RDY         ==^X81                  ; Controller and Drive ready bits
00000100                D51             ==^x100                 ; Drive 1 select
00000000                IDC_HLD::       .LONG  0                ; Temporary storage
```

```
                        00000000  028E   267 IDC_TST::       .LONG  0                    ; Temporary storage
                                  0292   268
                                  0292   269 ;
                                  0292   270 ;
                                  0292   271 ;
                             01FC 0292   272 .ENTRY UD_DRIVER1, ^M<R2,R3,R4,R5,R6,R7,R8>          ; UDA50 device driver.
                                  0294   273
                                  0294   274 ;
                                  0294   275 ; Start out by re-initing the UDA if it hasn't been inited yet,
                                  0294   276 ; or if there was an error previously.
                                  0294   277 ;
00000000'EF   00000000'EF    DE   0294   278         MOVAL   UNEX_E_I,SCB4               ; UNEXPECTED MACHINE CHECK HANDLER
         59   00000000'EF    9E   029F   279         MOVAB   RPB_BUFF,R9                 ; ADDRESS OF RPB
         53   00F26000 8F    D0   02A6   280         MOVL    #<^XF26000>,R3              ; BASE ADDRESS OF 11/730 UNIBUS
                                  02AD   281                                             ; ADAPTER
               5C A9    53    D0   02AD   282         MOVL    R3,RPB$L_ADPPHY(R9)         ; SET ADAPTER ADDRESS
               60 A9    53    D0   02B1   283         MOVL    R3,RPB$L_ADPVIR(R9)         ; SET ADAPTER ADDRESS
                                  02B5   284
         57   40FFF468 8F    D0   02B5   285         MOVL    #<^X40FFF468>,R7            ; FIXED ADDRESS OF UDA50
                                  02BC   286                                             ; AND LESI CONTROLLERS
               54 A9    57    D0   02BC   287         MOVL    R7,RPB$L_CSRPHY(R9)         ; SET CSR ADDRESS
               58 A9    57    D0   02C0   288         MOVL    R7,RPB$L_CSRVIR(R9)         ; SET CSR ADDRESS
                                  02C4   289
               03 FDE8 CF    E9   02C4   290         Blbc    W^Init, 3$                  ; Branch if already set
                     00E0    31   02C9   291         BRW     TEST_7
            FD2F CF    00    FB   02CC   292 3$:     CallS   #0, Ud_Init                 ; Call init code
                  0A 50    E8   02D1   293         BlbS    R0, 5$
00000000'EF    01    D0   02D4   294         MOVL    #1,NOCNTLR_FND
               FD22'    31   02DB   295         BRW     ERROR_ROUTINE               ; Exit if error
      FDCD CF    01    90   02DE   296 5$:     MovB    #1, W^Init                  ; Set init byte
                                  02E3   297
                                  02E3   298
```

```
                                02E3   300 .SBTTL TEST 6 DETERMINING CRD BASE CONTROLLER
                                02E3   301 ;++
                                02E3   302 ;       TEST DESCRIPTION:
                                02E3   303 ;
                                02E3   304 ;       THIS TEST DETERMINES THE BASE SUPPORTED CRD CONTROLLER.
                                02E3   305 ;
                                02E3   306 ;--
                                02E3   307
                                02E3   308 TEST_6::
00000000'EF   00000000'EF   DE  02E3   309          MOVAL   MCHK_IDC,SCB4           ; SET UP MACHINE CHECK
                                02EE   310                                          ; HANDLER FOR IDC
                                02EE   311 IDC:
00000000'9F   00000100 8F   D0  02EE   312          MOVL    #DS1,@#IDC_CSR          ; Drive 1 select
0000028A'9F   00000000'9F   D0  02F9   313          MOVL    @#IDC_CSR,@#IDC_HLD     ; Move status to temp storage
0000028E'9F   0000028A'9F   9A  0304   314          MOVZBL  @#IDC_HLD,@#IDC_TST     ; get rid of unnecessary bits
      0000028E'9F   81 8F   91  030F   315          CMPB    #C_D_RDY,@#IDC_TST      ; Controller and Drive ready set?
                      03    12  0317   316          BNEQ    1$                      ; No goto 1$
                    C040    31  0319   317          BRW     IDC_PARAM_1             ; Yes Boot RL02
                    0065    31  031C   318 1$:      BRW     IDC_PARAM_0             ; No Boot R80 System Disk
                                031F   319
                                031F   320 UDA_LESI::
00000000'EF   00000000'EF   9E  031F   321          MOVAB   MCHK_LES_UDA,SCB4               ; SET UP MACHINE CHECK
                                032A   322                                                  ; HANDLER FOR KLESI AND
                                032A   323                                                  ; UDA50
                                032A   324                                                  ;
        50    00000000'9F   B0  032A   325          MOVW    @#UDA_LESI_CSR,R0               ; SET UP CSR OF KLESI AND
                                0331   326                                                  ; UDA50 ON 730 UNIBUS
                                0331   327                                                  ; IF EITHER KLESI OR
                                0331   328                                                  ; UDA50 THERE....
                                0331   329
    FF5C CF     00      FB  0331   330          CALLS   #0,UD_DRIVER1                   ; CALL UDA50 & LESI
                                0336   331                                                  ; INITIALIZATION AND
                                0336   332                                                  ; STATUS
                                0336   333
                                03_6   334 LESI_UDA_PARAM::                         ; BOOT UDA OR LESI
                                0336   335                                          ; DRIVE TO BE DETERMINED
        50      11      D0  0336   336          MOVL    #^X11,R0                ; D/G/L 0 11
        51      03      D0  0339   337          MOVL    #3,R1                   ; D/G 1 3
52    0003F468 8F       D0  033C   338          MOVL    #^X03F468,R2            ; D/G 2 3F468
    53    FF39 CF       D0  0343   339          MOVL    WHAT_DRIVE,R3           ; R3 = DRIVE 0 OR 1
                    54    D4  0348   340          CLRL    R4                      ; D/G 4 0
55    00000000'EF       D0  034A   341          MOVL    WHAT_MODE,R5            ; R5 = CRD MODE
                    5D    D4  0351   342          CLRL    FP                      ; D/G D 0
5E    00000200 8F       D0  0353   343          MOVL    #^X200,SP               ; D/G E 200
                    6E    17  035A   344          JMP     (SP)
                                035C   345
                                035C   346
                                035C   347 IDC_PARAM_1::                            ; BOOT IDC RL02 CRDPACK
                                035C   348                                          ; DRIVE 1 - SQ1B00.CMD
50    00A80003 8F       D0  035C   349          MOVL    #^X0A80003,R0           ; D/G/L 0 00A80003
        51      03      D0  0363   350          MOVL    #3,R1                   ; D/G 1 3
52    00FFFB86 8F       D0  0366   351          MOVL    #^X0FFFB86,R2           ; D/G 2 FFFB86
        53      01      D0  036D   352          MOVL    #1,R3                   ; D/G 3 1
                    54    D4  0370   353          CLRL    R4                      ; D/G 4 0
55    00000000'EF       D0  0372   354          MOVL    WHAT_MODE,R5            ; D/G 5 8010
                    5D    D4  0379   355          CLRL    FP                      ; D/G D 0
5E    00000200 8F       D0  0378   356          MOVL    #^X200,SP               ; D/G E 200
```

```
                    6E   17   0382   357          JMP     (SP)
                                0384   358
                                0384   359 IDC_PARAM_0::                       ; BOOT IDC R80 CRDPACK
                                0384   360                                     ; DRIVE 0 - SQ0800.CMD
 50   00A80003      8F   D0   0384   361          MOVL    #^X0A80003,R0       ; D/G/L 0 00A80003
      51            03   D0   038B   362          MOVL    #3,R1               ; D/G 1 3
 52   00FFFB86      8F   D0   038E   363          MOVL    #^XUFFFB86,R2       ; D/G 2 FFFB86
      53            00   D0   0395   364          MOVL    #0,R3               ; D/G 3 0 Drive No. ''0''
                    54   D4   0398   365          CLRL    R4                  ; D/G 4 0
 55   00000000'     EF   D0   039A   366          MOVL    WHAT_MODE,R5        ; D/G 5 8010 Crd Mode
                    5D   D4   03A1   367          CLRL    FP                  ; D/G D 0
 5E   00000200      8F   D0   03A3   368          MOVL    #^X200,SP           ; D/G E 200
                    6E   17   03AA   369          JMP     (SP)
                                03AC   370
                                03AC   371
```

```
                        03AC   373 .SBTTL TEST 7 SIZE CRD DISK OPTIONS AND CHECK DRIVE STATUS
                        03AC   374 ;++
                        03AC   375 ; TEST DESCRIPTION:
                        03AC   376 ;
                        03AC   377 ; Bring the device on-line and check status
                        03AC   378 ;
                        03AC   379 ;--
                        03AC   380 TEST_7::
          64 A9   01    D0 03AC   381          MOVL    #1,RPB$W_UNIT(R9)          ; TEST UNIT 1 FIRST
      FECB CF   01    D0 03B0   382          MOVL    #1,WHAT_DRIVE
                        03B5   383 DRIVE0::
           FECD CF    94 03B5   384          CLRB    NODRIVE_0_1
           FECC CF    94 03B9   385          CLRB    SPIN_DWN
           FEC7 CF    94 03BD   386          CLRB    BAD_DISK
     55    FE57 CF    9E 03C1   387          MOVAB   W^CMDPKT, R5               ; Get the address of command packet
           85   01    D0 03C6   388          MOVL    #1,(R5)+                   ; Set command ref number
        85    64 A9   9A 03C9   389          MOVZBL  RPB$W_UNIT(R9),(R5)+       ; Put unit number in cmd packet field
           85   09    9A 03CD   390          MOVZBL  #MSCP$K_OP_ONLIN,(R5)+     ; Set opcode to bring drive online
                85    7C 03D0   391          CLRQ    (R5)+                      ; Clear byte count, buff desc
                65    7C 03D2   392          CLRQ    (R5)                       ;  buff desc and LBN
                   04CB   30 03D4   393          BSBW    IO                         ; Send it out
                          03D7   394
                52    D4 03D7   395          CLRL    R2
     52    FE7D CF    30 03D9   396          MOVW    RSPPKT+MSCP$W_STATUS,R2    ; CHECK STATUS
00000000'EF    52    91 03DE   397          CMPB    R2,MSCP$K_ST_SUCC          ; IF SUCCESS
                11    13 03E5   398          BEQL    NORMAL
           52   23    B1 03E7   399          CMPW    #^X23,R2                   ; IF NOT SPINNING
                07    12 03EA   400          BNEQ    7$                         ;
     FE98 CF    01    90 03EC   401          MOVB    #1,SPIN_DWN                ; SET A BIT TO TST LATER
                05    11 03F1   402          BRB     NORMAL
                          03F3   403
     FE90 CF    01    90 03F3   404 7$:      MOVB    #1,BAD_DISK                ; DRIVE OR CONTROLLER
                          03F8   405                                           ; ERROR
                          03F8   406
                          03F8   407 NORMAL:
                52    D4 03F8   408          CLRL    R2
     52    FE6E CF    D0 03FA   409          MOVL    RSPPKT+MSCP$Q_UNIT_ID+8,R2 ; UDA50 DISK TYPE
52    22A4103C 8F    D1 03FF   410          CMPL    #^X22A4103C,R2             ; RA60 ?
                03    12 0406   411          BNEQ    20$
           FBF5'      31 0408   412          BRW     RA60_TEST                  ; IF YES PRINT MESSAGE
                          0408   413
52    25641050 8F    D1 0408   414 20$:     CMPL    #^X25641050,R2             ; RA80 ?
                03    12 0412   415          BNEQ    30$
           FBE9'      31 0414   416          BRW     RA8081_TEST
                          0417   417
52    25641051 8F    D1 0417   418 30$:     CMPL    #^X25641051,R2             ; RA81 ?
                08    12 041E   419          BNEQ    40$
     FE5F CF    01    90 0420   420          MOVB    #1,RA81_PRES
           FBD8'      31 0425   421          BRW     RA8081_TEST
                          0428   422
52    20643019 8F    D1 0428   423 40$:     CMPL    #^X20643019,R2             ; RC25 ?
                08    12 042F   424          BNEQ    50$
     FE4F CF    01    90 0431   425          MOVB    #1,RC25_PRES
           FBC7'      31 0436   426          BRW     RC25_TEST
                          0439   427
           FE43 CF    D5 0439   428 50$:     TSTL    WHAT_DRIVE
                08    12 043D   429          BNEQ    INIT_DRV0                  ; NO DRIVE 1 TRY DRIVE 0
```

```
                FE42 CF    01    90  043F   430          MOVB    #1,NODRIVE_0_1          ; NO SUPPORTED DRIVE OPTION FOUND
                       FBB9'   31  0444   431          BRW     ERROR_ROUTINE           ; DRIVE 0 OR 1
                               0447   432
                               0447   433
                               0447   434  INIT_DRV0::
                  64 A9    00    D0  0447   435          MOVL    #0,RPB$W_UNIT(R9)       ; TEST UNIT 0
                FE30 CF    00    D0  044B   436          MOVL    #0,WHAT_DRIVE
                       FF62    31  0450   437          BRW     DRIVE0                  ; NOT A CRD SUPPORTED
                               0453   438                                              ; DISK OPTION TRY DRIVE 0
                               0453   439
```

```
                                        0453   441 .SBTTL TEST 8 RA60 AND RC25 VERIFY PACK LABEL CRDPACK
                                        0453   442 ;++
                                        0453   443 ;  TEST DESCRIPTION:
                                        0453   444 ;
                                        0453   445 ; READS RA60 (DRIVE 0 OR 1) AND RC25 (DRIVE 0) HOME BLOCK TO DETERMINE IF THE
                                        0453   446 ; PACK LABEL IS CORRECT AND THE HOME BLOCK IS VALID.
                                        0453   447 ;
                                        0453   448 ;--
                                        0453   449 .ALIGN  PAGE
                                        0600   450
                            00000000    0600   451 HMBLK_BUFF:      .LONG   0
                            00000000    0604   452 VALID1:          .LONG   0
                            00000000    0608   453 VALID2:          .LONG   0
                            000007D8    060C   454                 .BLKL   115
                            00000000    07D8   455 CRDP:            .LONG   0
                            00000000    07DC   456 ACK_:            .LONG   0
                            00000800    07E0   457                 .BLKL   8
                                        0800   458
                                        0800   459 TEST_8::
                     FA83 CF    94      0800   460         CLRB    HMBLK_ERR
                          51    D4      0804   461         CLRL    R1
                53   00F26000 8F    D0  0806   462         MOVL    #<^XF26000>,R3
                     00000021'EF    D0  080D   463         MOVL    #1,R5                   ; LBN = 1
                58   00000200 8F    D0  0810   464         MOVL    #512,R8                 ; BYTE COUNT = 512
             10 AC   00000021'EF    D0  0817   465         MOVL    IO$_READLBLK,FUNC(AP)   ; IO OPERATION
                     FA00 CF    21  90  081F   466         MOVB    #MSCP$K_OP_READ, -      ; READ COMMAND
                                        0824   467                 CMDPKT+MSCP$B_OPCODE
                     FA0F CF    55  D0  0824   468         MOVL    R5,CMDPKT+MSCP$L_LBN    ; SET THE LOGICAL BLOCK NUMBER
                     F9FA CF    58  D0  0829   469         MOVL    R8,CMDPKT+MSCP$L_BYTE_CNT ; SET THE BYTE COUNT
                                        082E   470 ;
                                        082E   471 ; SETTING UP 2 UNIBUS MAPPING REGISTERS AND HOME BLOCK BUFFER
                                        082E   472 ;
                5A      FDCE CF    DE   082E   473         MOVAL   HMBLK_BUFF,R10                  ; ADDRESS OF TRANSFER BUFFER
             51 5A      15   09    EF   0833   474         EXTZV   #VA$V_VPN,#VA$S_VPN,R10,R1       ; GET PFN
                54      0E40 C3    DE   0838   475         MOVAL   UBA$L_MAP+<400*4>(R3),R4        ; POINT TO UNIBUS MAP
                                        083D   476                                                 ; REGISTER 400
          84    51       80000000 8F    C9   083D   477         BISL3   #UBA$M_BRRVR_AIR,R1,(R4)+       ; SET AS VALID W/PFN
                                   51   D6   0845   478         INCL    R1                              ; NEXT PFN
          64    51       80000000 8F    C9   0847   479         BISL3   #UBA$M_BRRVR_AIR,R1,(R4)        ; SET AS VALID W/PFN
    5A    15    09       00000190 8F    F0   084F   480         INSV    #400,#VA$V_VPN,#VA$S_VPN,R10    ; SET MAP REGISTER TO US.
                     F9CF CF    5A  D0   0858   481         MOVL    R10,CMDPKT+MSCP$B_BUFFER        ; SET THE UNIBUS MAP REGISTER
                                        085D   482
                                   43   10   085D   483         BSBB    IO                      ;
                                        085F   484
                          FDA1 CF    D5   085F   485         TSTL    VALID1                  ; TEST IF THE HOME BLOCK IS VALID
                                   07   12   0863   486         BNEQ    V_HMBLK                 ; IF EQ 0 THEN POSSIBLY CONTROLLER,
                     FA1D CF    01   90   0865   487         MOVB    #1,HMBLK_ERR            ; DRIVE OR DISK ERROR
                                   26   11   086A   488         BRB     PREP_ERR
                                        086C   489 V_HMBLK:
                          FD98 CF    D5   086C   490         TSTL    VALID2
                                   07   12   0870   491         BNEQ    LABEL
                     FA10 CF    01   90   0872   492         MOVB    #1,HMBLK_ERR
                                   19   11   0877   493         BRB     PREP_ERR
                                        0879   494 LABEL:
          FF56 CF    50445243 8F    D1   0879   495         CMPL    #CRDPACK1,CRDP          ; COMPARE LABEL
                                   0E   12   0882   496         BNEQ    PREP_ERR                ; IF NOT CORRECT
          FF4F CF    204B4341 8F    D1   0884   497         CMPL    #CRDPACK2,ACK_          ; PRINT ERROR PREP
```

```
                    03   12  088D   498          BNEQ    PREP_ERR
                  FAA4   31  088F   499          BRW     LESI_UDA_PARAM
                         0892   500
                         0892   501 PREP_ERR:
          F9F2 CI   01   90  0892   502          MOVB    #1,SPIN_DWN             ; PREPARATION ERROR WRONG PACK
            03 F9EA CF   E8  0897   503          BLBS    RC25_PRES,20$           ; TEST IF RC25
                  F761'  31  089C   504          BRW     RA60_TEST
                  F75E'  31  089F   505 20$:     BRW     RC25_TEST
                         08A2   506
                         08A2   507
               54   02 A7  B0  08A2   508 10:     MOVW    UDASA(R7),R4            ; Controller offline?
                    27  12  08A6   509          BNEQ    ERROR1                  ; Yep, error out
          F967 CF   8000 8F  A8  08A8   510          BISW    #OWN, CD+2              ; Set controller ownership
          F95C CF   8000 8F  A8  08AF   511          BISW    #OWN, RD+2              ; Ditto
               54   67  B0  08B6   512          MOVW    UDAIP(R7),R4            ; Tell controller to read
               54   02 A7  B0  08B9   513 30$:     MovW    UdaSa(R7), R4           ; Any problems?
                    10  12  08BD   514          BNeq    Error1                  ; Yes
                F94F CF  B5  08BF   515          TSTW    RD+2                    ; Any response back?
                    F4  19  08C3   516          BLSS    30$                     ; No, spin until there is
                F991 CF  B5  08C5   517          TSTW    RSPPKT+MSCP$W_STATUS    ; Any drive errors?
                    04  12  08C9   518          BNEQ    ERROR1                  ; Yes
                        08CB   519
                        08CB   520 ;
                        08CB   521 ; Transfer is complete. Return with success status code.
                        08CB   522 ;
                50  01  3C  08CB   523          MOVZWL  #SS$_NORMAL,R0          ; SET COMPLETION CODE
                    05  08CE   524          RSB                             ; AND RETURN
                        08CF   525 ;
                        08CF   526 ; Error occured during transfer.  Return and retry.
                        08CF   527 ;
                        08CF   528
                        08CF   529 ERROR1:
                F7DD CF  94  08CF   530          ClrB    W^Init                  ; Must re-init next time
          50    0054 8F  3C  08D3   531          MOVZWL  #SS$_CTRLERR,R0         ; Set failure status
                    05  08D8   532          RSB                             ; Return to BOOTDRIVR
                        08D9   533
                        08D9   534
                        08D9   535          .END
```

```
ACK                             000007DC R      02
BAD_DISK                        00000288 RG     02
BYTE_OFF                        00000077 R      02
CD                              00000214 R      02
CMDINT                          0000020C R      02
CMDPKT                          0000021C R      02
CRDP                            000007D8 R      02
CRDPACK1                    =   50445243
CRDPACK2                    =   20484341
C_D_RDY                     =   00000081   G
DRIVE0                          000003B5 RG     02
DS$GB_UDA50_INIT                000000B0 RG     02
DS1                         =   00000100   G
ERROR                           000000A2 R      02
ERROR1                          000008CF R      02
ERROR_ROUTINE                   ********     X  02
FUNC                        =   00000010
GO                          =   00000001
HMBLK_BUFF                      00000600 R      02
HMBLK_ERR                       00000287 RG     02
IDC                             000002EE R      02
IDC_CSR                         ********     X  02
IDC_HLD                         0000028A RG     02
IDC_PARAM_0                     00000384 RG     02
IDC_PARAM_1                     0000035C RG     02
IDC_TST                         0000028E RG     02
INIT                            000000B0 R      02
INIT_DRV0                       00000447 RG     02
INTTBL                          00000200 R      02
IO                              000008A2 R      02
IO$_READLBLK                =   00000021
LABEL                           00000879 R      02
LESI_UDA_PARAM                  00000336 RG     02
LOOP                            00000063 R      02
MCHK_IDC                        ********     X  02
MCHK_LES_UDA                    ********     X  02
MODE                        =   00000014
MSCP$B_BUFFER               =   00000010
MSCP$B_OPCODE               =   00000008
MSCP$K_OP_ONLIN             =   00000009
MSCP$K_OP_READ              =   00000021
MSCP$K_ST_SUCC              =   00000000
MSCP$L_BYTE_CNT             =   0000000C
MSCP$L_LBN                  =   0000001C
MSCP$Q_UNIT_ID              =   00000014
MSCP$W_STATUS               =   0000000A
NOCNTLR_FND                     ********     X  02
NODRIVE_0_1                     00000286 RG     02
NORMAL                          000003F8 R      02
OWN                         =   00008000
PR$_MAPEN                   =   00000038
PR730$_TODR                 =   0000001B
PREP_ERR                        00000892 R      02
PTE$M_PFN                   =   001FFFFF
RA60_TEST                       ********     X  02
RA8081_TEST                     ********     X  02
RA81_PRES                       00000284 RG     02
RC25_PRES                       00000285 RG     02
RC25_TEST                       ********     X  02
RD                              00000210 R      02
RETRY                           00000051 R      02
RING                            00000210 R      02
RPB$L_ADPPHY                =   0000005C
RPB$L_ADPVIR                =   00000060
RPB$L_CSRPHY                =   00000054
RPB$L_CSRVIR                =   00000058
RPB$L_SVASPT                =   00000050
RPB$W_UNIT                  =   00000064
RPB_BUFF                        ********     X  02
RSPINT                          0000020E R      02
RSPPKT                          00000250 R      02
S1                          =   0000000B
S4                          =   0000000E
SCB4                            ********     X  02
SPIN_DWN                        00000289 RG     02
SS$_CTRLERR                 =   00000054
SS$_NORMAL                  =   00000001
TEST_6                          000002F3 RG     02
TEST_7                          000003AC RG     02
TEST_8                          00000800 RG     02
TIME                            0000005B R      02
UBA$L_MAP                   =   00000800
UBA$M_BRRVR_AIR             =   80000000
UDAIP                       =   00000000
UDASA                       =   00000002
UDA_LESI                        0000031F RG     02
UDA_LESI_CSR                    ********     X  02
UDA_RESPONSE                    00000250 RG     02
UD_DRIVER1                      00000292 RG     02
UD_INIT                         00000000 RG     02
UMR                         =   000001EE
UNEX_E_I                        ********     X  02
VA$S_VPN                    =   00000015
VA$V_VPN                    =   00000009
VALID                           000000AC R      02
VALID1                          00000604 R      02
VALID2                          00000608 R      02
V_HMBLK                         0000086C R      02
WHAT_DRIVE                      00000280 RG     02
WHAT_MODE                       ********     X  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
SIZING	000008D9 (2265.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC PAGE

Symbol Cross Reference

SYMBOL	VALUE	DEFINITION	REFERENCES...
ACK_	000007DC-R	456 (3)	#-497 (3)
BAD_DISK	00000288-R	258 (3)	#-386 (3) #-404 (3)
BYTE_OFF	00000077-R	155 (2)	#-119 (2) #-165 (2) #-168 (2)
CD	00000214-R	199 (2)	#-169 (2) #-510 (3)
CMDINT	0000020C-R	195 (2)	
CMDPKT	0000021C-R	203 (2)	167 (2) 387 (3) #-467 (3) #-468 (3) #-469 (3) #-481 (3)
CRDP	000007D8-R	455 (3)	#-495 (3)
CRDPACK1	=50445243	250 (3)	#-495 (3)
CRDPACK2	=204B4341	251 (3)	#-497 (3)
C_D_RDY	=00000081	264 (3)	#-315 (3)
DRIVE0	000003B5-R	383 (3)	#-437 (3)
DS$GB_UDA50_INIT	000000B0-R	180 (2)	
DS1	=00000100	265 (3)	#-312 (3)
ERROR	000000A2-R	172 (2)	#-148 (2) #-152 (2)
ERROR1	000008CF-R	529 (3)	#-509 (3) #-514 (3) #-518 (3)
ERROR_ROUTINE	00000000-XR		#-295 (3) #-431 (3)
FUNC	=00000010	247 (3)	#-465 (3)
GO	=00000001	76 (2)	192 (2)
HMBLK_BUFF	00000600-R	451 (3)	473 (3)
HMBLK_ERR	00000287-R	257 (3)	#-460 (3) #-487 (3) #-492 (3)
IDC	000002EE-R	311 (3)	
IDC_CSR	00000000-XR		#-312 (3) #-313 (3)
IDC_HLD	0000028A-R	266 (3)	#-313 (3) #-314 (3)
IDC_PARAM_0	00000384-R	359 (3)	#-318 (3)
IDC_PARAM_1	0000035C-R	347 (3)	#-317 (3)
IDC_TST	0000028E-R	267 (3)	#-314 (3) #-315 (3)
INIT	000000B0-R	181 (2)	#-290 (3) #-296 (3) #-530 (3)
INIT_DRV0	00000447-R	434 (3)	#-429 (3)
INTTBL	00000200-R	190 (2)	118 (2) #-121 (2) 139 (2)
IO	000008A2-R	508 (3)	#-393 (3) #-483 (3)
IO$_READLBLK	=00000021		#-465 (3)
LABEL	00000879-R	494 (3)	#-491 (3)
LESI_UDA_PARAM	00000336-R	334 (3)	#-499 (3)
LOOP	00000063-R	147 (2)	#-153 (2)
MCHK_IDC	00000000-XR		309 (3)
MCHK_LES_UDA	00000000-XR		321 (3)
MODE	=00000014	248 (3)	
MSCP$B_BUFFER	=00000010		#-481 (3)
MSCP$B_OPCODE	=00000008		#-467 (3)
MSCP$K_OP_ONLIN	=00000009		#-390 (3)
MSCP$K_OP_READ	=00000021		#-466 (3)
MSCP$K_ST_SUCC	=00000000		#-397 (3)
MSCP$L_BYTE_CNT	=0000000C		#-469 (3)
MSCP$L_LBN	=0000001C		#-468 (3)
MSCP$Q_UNIT_ID	=00000014		#-409 (3)
MSCP$W_STATUS	=0000000A		#-396 (3) #-517 (3)
NOCNTLR_FND	00000000-XR		#-294 (3)
NODRIVE_0_1	00000286-R	256 (3)	#-384 (3) #-430 (3)
NORMAL	000003F8-R	407 (3)	#-398 (3) #-402 (3)

```
OWN                     =00008000      77  (2)     190  (2)    #-510  (3)    #-511  (3)
PR$_MAPEN               =00000038                 #-110  (2)
PR730$_TODR             =0000001B                 #-145  (2)    #-150  (2)
PREP_ERR                 00000892-R    501 (3)    #-488  (3)    #-493  (3)    #-496  (3)    #-498  (3)
PTE$M_PFN               =001FFFFF                 #-129  (2)
RA60_TEST                00000000-XR              #-412  (3)    #-504  (3)
RA8081_TEST              00000000-XR              #-416  (3)    #-421  (3)
RA81_PRES                00000284-R    254 (3)    #-420  (3)
RC25_PRES                00000285-R    255 (3)    #-425  (3)    #-503  (3)
RC25_TEST                00000000-XR              #-426  (3)    #-505  (3)
RD                       00000210-R    198 (2)    #-166  (2)    #-511  (3)    #-515  (3)
RETRY                    00000051-R    139 (2)    #-172  (2)
RING                     00000210-R    197 (2)    #-121  (2)
RPB$L_ADPPHY            =0000005C                   123  (2)    #-124  (2)    #-282  (3)
RPB$L_ADPVIR            =00000060                   123  (2)    #-283  (3)
RPB$L_CSRPHY            =00000054                   125  (2)    #-126  (2)    #-287  (3)
RPB$L_CSRVIR            =00000058                   125  (2)    #-288  (3)
RPB$L_SVASPT            =00000050                 #-128  (2)
RPB$W_UNIT              =00000064                 #-381  (3)    #-389  (3)    #-435  (3)
RPB_BUFF                 00000000-XR                279  (3)
RSPINT                   0000020E-R    196 (2)
RSPPKT                   00000250-R    208 (2)      164  (2)    #-396  (3)    #-409  (3)    #-517  (3)
S1                      =0000000B      78  (2)    #-140  (2)
S4                      =0000000E      79  (2)    #-159  (2)
SCB4                     00000000-XR              #-278  (3)    #-309  (3)    #-321  (3)
SPIN_DWN                 00000289-R    259 (3)    #-385  (3)    #-401  (3)    #-502  (3)
SS$_CTRLERR             =00000054                 #-173  (2)    #-531  (3)
SS$_NORMAL              =00000001                 #-523  (3)
TEST_6                   000002E3-R    308 (3)
TEST_7                   000003AC-R    380 (3)    #-291  (3)
TEST_8                   00000800-R    459 (3)
TIME                     0000005B-R    145 (2)    #-159  (2)
UBA$L_MAP               =00000800                   130  (2)      475  (3)
UBA$M_BRRVR_AIR         =80000000                 #-477  (3)    #-479  (3)
UDAIP                   =00000000      74  (2)    #-141  (2)    #-512  (3)
UDASA                   =00000002      75  (2)    #-147  (2)    #-158  (2)    #-508  (3)    #-513  (3)
UDA_LESI                 0000031F-R    320 (3)
UDA_LESI_CSR             00000000-XR              #-325  (3)
UDA_RESPONSE             00000250-R    207 (2)
UD_DRIVER1               00000292-R    272 (3)      330  (3)
UD_INIT                  00000000-R    106 (2)      292  (3)
UMR                     =000001EE      80  (2)    #-117  (2)      130  (2)
UNEX_E_I                 00000000-XR                278  (3)
VA$S_VPN                =00C00015                   #-122  (2)    #-474  (3)    #-480  (3)
VA$V_VPN                =00000009                   #-122  (2)    #-474  (3)    #-480  (3)
VALID                    000000AC-R    178 (2)    #-131  (2)    #-133  (2)
VALID1                   00000604-R    452 (3)    #-485  (3)
VALID2                   00000608-R    453 (3)    #-490  (3)
V_HMBLK                  0C000860-R    489 (3)    #-486  (3)
WHAT_DRIVE               00000280-R    253 (3)    #-339  (3)    #-382  (3)    #-428  (3)    #-436  (3)
WHAT_MODE                00000000-XR              #-341  (3)    #-354  (3)    #-366  (3)
```

Macros Cross Reference

MACRO	SIZE	DEFINITION		REFERENCES...									
$BTDDEF	1	59	(2)	59	(2)								
$DEFINI	1	59	(2)	59	(2)	60	(2)	61	(2)	62	(2)	63	(2)
				64	(2)	65	(2)	66	(2)	67	(2)	68	(2)
				69	(2)								
$IODEF	1?	60	(2)	60	(2)								
$MSCPDEF	18	61	(2)	61	(2)								
$PR730DEF	1	63	(2)	63	(2)								
$PRDEF	4	62	(2)	62	(2)								
$PTEDEF	3	64	(2)	64	(2)								
$RPBDEF	5	65	(2)	65	(2)								
$SSDEF	21	66	(2)	66	(2)								
$UBADEF	6	67	(2)	67	(2)								
$UBIDEF	2	68	(2)	68	(2)								
$VADEF	1	69	(2)	69	(2)								
ASSUME	1	123	(2)	123	(2)	125	(2)						

Opcode Cross Reference

OPCODE	VALUE	REFERENCES...													
ADDL	00C0	121	(2)												
AOBLEQ	00F3	159	(2)												
BBS	00E0	149	(2)												
BEQL	0013	398	(3)												
BGTRU	001A	152	(2)												
BICL	00CA	129	(2)	165	(2)	168	(2)								
BICL3	00CB	119	(2)												
BISL3	00C9	120	(2)	131	(2)	133	(2)	166	(2)	169	(2)	477	(3)	479	(3)
BISW	00A8	510	(3)	511	(3)										
BLBC	00E9	127	(2)	290	(3)										
BLBS	00E8	293	(3)	503	(3)										
BLSS	0019	148	(2)	516	(3)										
BNEQ	0012	316	(3)	400	(3)	411	(3)	415	(3)	419	(3)	424	(3)	429	(3)
		486	(3)	491	(3)	496	(3)	498	(3)	509	(3)	514	(3)	518	(3)
BRB	0011	153	(2)	402	(3)	488	(3)	493	(3)						
BRW	0031	291	(3)	295	(3)	317	(3)	318	(3)	412	(3)	416	(3)	421	(3)
		426	(3)	431	(3)	437	(3)	499	(3)	504	(3)	505	(3)		
BSBB	0010	483	(3)												
BSBW	0030	393	(3)												
CALLS	00FB	292	(3)	330	(3)										
CLRB	0094	384	(3)	385	(3)	386	(3)	460	(3)	530	(3)				
CLRL	00D4	340	(3)	342	(3)	353	(3)	355	(3)	365	(3)	367	(3)	395	(3)
		408	(3)	461	(3)										
CLRQ	007C	391	(3)	392	(3)										
CLRW	00B4	141	(2)												
CMPB	0091	315	(3)	397	(3)										
CMPL	00D1	151	(2)	410	(3)	414	(3)	418	(3)	423	(3)	495	(3)	497	(3)
CMPW	00B1	399	(3)												
EXTZV	00EF	122	(2)	474	(3)										
INCL	00D6	132	(2)	478	(3)										
INSV	00F0	480	(3)												
JMP	0017	344	(3)	357	(3)	369	(3)								
MFPR	00DB	110	(2)	145	(2)	150	(2)								
MOVAB	009E	118	(2)	139	(2)	146	(2)	164	(2)	167	(2)	279	(3)	321	(3)
		387	(3)												
MOVAL	00DE	130	(2)	278	(3)	309	(3)	473	(3)	475	(3)				
MOVB	0090	296	(3)	401	(3)	404	(3)	420	(3)	425	(3)	430	(3)	466	(3)
		487	(3)	492	(3)	502	(3)								
MOVL	00D0	117	(2)	124	(2)	126	(2)	128	(2)	138	(2)	140	(2)	280	(3)
		282	(3)	283	(3)	285	(3)	287	(3)	288	(3)	294	(3)	312	(3)
		313	(3)	336	(3)	337	(3)	338	(3)	339	(3)	341	(3)	343	(3)
		349	(3)	350	(3)	351	(3)	352	(3)	354	(3)	356	(3)	361	(3)
		362	(3)	363	(3)	364	(3)	366	(3)	368	(3)	381	(3)	382	(3)
		388	(3)	409	(3)	435	(3)	436	(3)	462	(3)	463	(3)	464	(3)
		465	(3)	468	(3)	469	(3)	481	(3)						
MOVW	00B0	147	(2)	158	(2)	325	(3)	396	(3)	508	(3)	512	(3)	513	(3)
MOVZBL	009A	314	(3)	389	(3)	390	(3)								
MOVZWL	003C	173	(2)	523	(3)	531	(3)								
RET	0004	170	(2)	174	(2)										
RSB	0005	524	(3)	532	(3)										

SOBGTR	00F5	172	(2)				
TSTL	00D5	428	(3)	485	(3)	490	(3)
TSTW	00B5	515	(3)	517	(3)		

Directives Cross Reference

DIRECTIVE	REFERENCES...															
.ALIGN	188	(2)	449	(3)												
.BLKL	454	(3)	457	(3)												
.BLKW	204	(2)	206	(2)	208	(2)										
.BYTE	182	(2)	254	(3)	255	(3)	256	(3)	257	(3)	258	(3)	259	(3)		
.DISABLE	176	(2)														
.ENABLE	108	(2)														
.END	535	(3)														
.ENDC	123	(2)	125	(2)												
.ENDM	123	(2)	59	(2)	60	(2)	61	(2)	62	(2)	63	(2)	64	(2)	65	(2)
	66	(2)	67	(2)	68	(2)	69	(2)								
.ENTRY	106	(2)	272	(3)												
.IDENT	2	(1)														
.IF	123	(2)	125	(2)												
.IFF	123	(2)	125	(2)												
.LONG	156	(2)	178	(2)	191	(2)	198	(2)	199	(2)	253	(3)	266	(3)	267	(3)
	451	(3)	452	(3)	453	(3)	455	(3)	456	(3)						
.NOCROSS	59	(2)	60	(2)	61	(2)	62	(2)	63	(2)	64	(2)	65	(2)	66	(2)
	67	(2)	68	(2)	69	(2)										
.PAGE	299	(3)	372	(3)	440	(3)										
.PSECT	90	(2)														
.RESTORE	59	(2)	60	(2)	61	(2)	62	(2)	63	(2)	64	(2)	65	(2)	66	(2)
	67	(2)	68	(2)	69	(2)										
.SAVE	59	(2)	60	(2)	61	(2)	62	(2)	63	(2)	64	(2)	65	(2)	66	(2)
	67	(2)	68	(2)	69	(2)										
.SBTTL	210	(3)	300	(3)	373	(3)	441	(3)	54	(2)	92	(2)				
.TITLE	1	(1)														
.WORD	190	(2)	192	(2)	194	(2)	195	(2)	196	(2)	201	(2)	202	(2)	203	(2)

Register Cross Reference

REGISTER	NO. REFERENCES	REFERENCES...
AP	1	#-465 (3)
FP	3	#-342 (3)
R0	15	#-110 (2) #-124 (2) #-126 (2) #-127 (2) #-140 (2) #-149 (2)
		#-159 (2) #-173 (2) #-293 (3) #-325 (3) #-336 (3) #-349 (3)
		#-361 (3) #-523 (3) #-531 (3)
R1	18	#-119 (2) #-120 (2) #-150 (2) #-151 (2) #-164 (2) #-165 (2)
		#-166 (2) #-167 (2) #-168 (2) #-169 (2) #-337 (3) #-350 (3)
		#-362 (3) #-461 (3) #-474 (3) #-477 (3) #-478 (3) #-479 (3)
R10	4	#-473 (3) 474 (3) 480 (3) #-481 (3)
R2	31	106 (2) #-118 (2) #-119 (2) #-120 (2) #-121 (2) 122 (2)
		#-128 (2) #-129 (2) #-131 (2) #-132 (2) #-133 (2) #-145 (2)
		146 (2) #-151 (2) 272 (3) #-338 (3) #-351 (3) #-363 (3)
		#-395 (3) #-396 (3) #-397 (3) #-399 (3) #-408 (3) #-409 (3)
		#-410 (3) #-414 (3) #-418 (3) #-423 (3)
R3	14	106 (2) #-124 (2) 130 (2) #-139 (2) #-158 (2) 272 (3)
		#-280 (3) #-282 (3) #-283 (3) #-339 (3) #-352 (3) #-364 (3)
		#-462 (3) 475 (3)
R4	16	106 (2) #-130 (2) #-131 (2) #-133 (2) #-147 (2) 149 (2)
		272 (3) #-340 (3) #-353 (3) #-365 (3) #-475 (3) #-477 (3)
		#-479 (3) #-508 (3) #-512 (3) #-513 (3)
R5	13	106 (2) 272 (3) #-341 (3) #-354 (3) #-366 (3) #-387 (3)
		#-388 (3) #-389 (3) #-390 (3) #-391 (3) #-392 (3) #-463 (3)
		#-468 (3)
R6	6	106 (2) #-117 (2) #-120 (2) #-166 (2) #-169 (2) 272 (3)
R7	12	106 (2) #-126 (2) #-141 (2) #-147 (2) #-158 (2) 272 (3)
		#-285 (3) #-287 (3) #-288 (3) #-508 (3) #-512 (3) #-513 (3)
R8	6	106 (2) #-138 (2) #-172 (2) 272 (3) #-464 (3) #-469 (3)
R9	11	#-124 (2) #-126 (2) #-128 (2) #-279 (3) #-282 (3) #-283 (3)
		#-287 (3) #-288 (3) #-381 (3) #-389 (3) #-435 (3)
SP	6	#-343 (3) 344 (3) #-356 (3) 357 (3) #-368 (3) 369 (3)

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	96	00:00:00.49	00:00:02.89
Command processing	143	00:00:00.70	00:00:04.11
Pass 1	1775	00:00:20.87	00:02:39.91
Symbol table sort	38	00:00:02.60	00:00:32.38
Pass 2	392	00:00:03.11	00:00:28.54
Symbol table output	40	00:00:00.11	00:00:00.95
Psect synopsis output	6	00:00:00.03	00:00:00.29
Cross-reference output	188	00:00:01.02	00:00:07.56
Assembler run totals	2683	00:00:28.95	00:03:56.65

The working set limit was 500 pages.
82237 bytes (161 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1624 non-local and 12 local symbols.

535 source lines were read in Pass 1, producing 22 object records in Pass 2.
60 pages of virtual memory were used to define 18 macros.

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

Macro library name	Macros defined
DRB1:[DS.WORK]DS.MLB;218	0
SYS$SYSROOT:[SYSLIB]LIB.MLB;1	7
SYS$SYSROOT:[SYSLIB]STARLET.MLB;1	8
TOTALS (all libraries)	15

1657 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

MACRO/OBJ/LIST/CROSS=ALL ENKAZ02A+SYS$LIBRARY:LIB/LIB+DRB1:[DS.WORK]DS.MLB/LIB