

PDP11-70/74

11/70 POW FAIL
CEKBGCO

AH-7984C-MC
FICHE 1 OF 1

MAY 1980
COPYRIGHT © 75, 80
MADE IN USA



IDENTIFICATION

PRODUCT CODE: AC-7983C-MC

PRODUCT NAME: CEKBGCO 11/70 Pow Fail

PRODUCT DATE: MAY, 1980

MAINTAINER: Diagnostic Engineering

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1975, 1980 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 Equipment
 - 2.2 Storage
 - 2.3 Preliminary Procedures
- 3.0 Loading Procedure
- 4.0 Starting Procedure
 - 4.1 Starting Address
 - 4.2 Control Switch Settings
 - 4.3 Restarting Procedure
 - 4.4 Program and/or Operator action.
- 5.0 Operating Procedures
 - 5.1 Modes of Operation
 - 5.2 Common Procedures
 - 5.3 Notes and Warnings
 - 5.4 Uniprocessor mode, no UBE
 - 5.5 Uniprocessor mode, with UBE
 - 5.6 Multiprocessor mode, no UBE
 - 5.7 Multiprocessor mode, with UBE
- 6.0 Errors and Error Reporting
- 7.0 Test Descriptions
- 8.0 Subroutine Abstracts
- 9.0 Restrictions
- 10.0 Miscellaneous

REVISION HISTORY

REVISION: MODIFIED TESTS 17,20 AND 21 TO UTILIZE 'ASRB' LOCKS IN ACCORDANCE
WITH SPECIFIED CONSTRUCTION.

1.0 ABSTRACT

This diagnostic serves as a replacement to DEKBGA, the 11/70 Power Fail Test. It provides all of the coverage and features of the previous power fail test along with the added features of diagnostic support for multiprocessor systems using the KB11-CM Processor and the IIST interface.

The test is divided into two sections, Section 1 and Section 2. Section 1 completely replaces DEKBGA and basically serves as a test of CPU logic validity during a power fail sequence. This section can be run on a standard unmodified 11/70 processor as it utilizes no multiprocessor resources. Section 2 is enabled by enabling console switch 6. This section will be executed directly after section 1. This section provides testing for proper functionality of the Interprocessor Interrupt and Sanity Timer (IIST) powerfail procedures, and memory system activity during a loss of power on the system or one of its subsystems.

This diagnostic also supports the use of the UNIBUS Exerciser (UBE) to perform simulated powerfail sequences during the execution of Section 1.

2.0 REQUIREMENTS

2.1 Equipment

THIS DIAGNOSTIC IS DESIGNED TO TEST PDP-11/70 SYSTEMS USING THE KB-11B/C PROCESSOR. A CONSOLE TERMINAL IS REQUIRED FOR MESSAGE AND ERROR REPORTING. A UNIBUS EXERCISER MAY OPTIONALLY BE USED.

NOTE: THIS DIAGNOSTIC ALSO SUPPORTS THE PDP-11/74, AN IN-HOUSE, EXPERIMENTAL PROCESSOR IN BOTH STANDALONE AND MP MODES.

2.2 Storage

The Power Fail test runs in 12K words of memory. The first 4K is used for stack areas and common access data. The second 4K contains the program itself. The only data elements that reside here are type and error messages, and execution flow and control variables that are used

(modified) when the processor runs in multiprocessor mode. The next 4K is used as buffer space for massbus transfers that occur during test 21.

2.3 Preliminary Procedures.

All standard 11/70-74 CPU diagnostics should first be run to insure proper operation under a secure power condition. Specifically, the IIST diagnostic and Unibus Exerciser diagnostics should be run. If massbus devices are to be used for test 21 it might be wise to insure that the devices are in proper working order. The power fail diagnostic may use all massbus drives and memory boxes accessible by the CPUs participating in this diagnostic. Therefore, it is important to switch offline to participating CPUs all data storage equipment that the operator does not want corrupted by this diagnostic. Before starting the power-fail diagnostic the operator should also be sure that the CPU power-up action switches (on the IIST front panel) are all 'Run or Halt' for participating CPUs, the IIST enable switches are 'online', and the IIST configuration switches are all at the same system for all participating CPUs. All participating memory boxes should be online to all participating CPUs.

The operator of this test should be familiar with the MKA11 and IIST boot/control panel.

3.0 LOADING PROCEDURE

This diagnostic can be loaded within the standard XXDP loading procedure. Note that if this diagnostic is being run as part of a chain that it must be the last element in that chain.

This diagnostic is loadable under the APT system but can only be run under APT in Uniprocessor mode with the UBE enabled.

4.0 STARTING PROCEDURE

4.1 Starting Address

The starting address for this diagnostic is 200. The restart address is 220.

4.2 Control Switch Settings

The switch settings are as follows (when set to 1):

- SW15 - Halt on Error
- SW14 - Loop on test (Section 1 only)
- SW08 - Enable System Power Fail Test (Test 25)
- SW07 - Disable Section 1 Tests (Multiprocessor mode only)

<u>Sw6</u>	<u>Sw5</u>	
0	0	Uniprocessor mode, manual powerfail. (this mode can be used as a replacement for previous powerfail diagnostic)
0	1	Uniprocessor mode, automatic (UBE) powerfail (uniprocessor mode that should be used under APT)
1	0	Multiprocessor mode, manual powerfail. Both sections (1 and 2) are executed.
1	1	Multiprocessor mode, automatic (UBE) powerfail. Only Section 1 will use the UBE. Each CPU must have a UBE.

Note that for the multiprocessor modes (SW6 enabled) the IIST 'system' ID of each CPU participating in the test should be specified by setting the appropriate bit in Switches 0-3 in the switch register. For example, if all processors in a 4 processor system are to be tested then switches 0-3 should be enabled. If only CPU's 0 and 2 are to participate then only switches 0 and 2 should be enabled.

The switches are defined by the master CPU only (in multiprocessor mode), where the master is the CPU that starts or restarts the program. Each slave CPU uses the switch definitions of the master.

The Program will not interpret changes to the switches in multiprocessor mode while the program is running. The program must be restarted.

4.3 Restarting Procedures

The restart address for this diagnostic is 220. A restart should be considered a completely new session therefore be sure the proper switches are enabled for the features that you want.

4.4 Program And/Or Operator Action

NOTE

Be sure to isolate the CPU's that are to participate in the powerfail test from other CPU's that are to remain functional. Participating CPU's should have the IIST configuration switches found on the IIST panel switched to the SYSTEM B position (if this is the default position for the installation then the SYSTEM A position could be used).

5.0 OPERATING PROCEDURES

5.1 Modes Of Operation

This diagnostic is designed to run in four basic modes specified by the selection of bits 5 & 6 in the console switch register. It should be noted that if a multiprocessor mode is chosen (SW6 enabled) that the first CPU started will become the 'master' and successively start the remaining participating CPU's, the 'Slaves'.

5.2 Common Procedures

Load address 200 and then enable the switches for the test mode that is desired. Remember to enable the appropriate CPU mask bit if you are running a multiprocessor mode. When this has been done insure that all slaves are powered up and have their halt switches in the 'enable' state if MP mode is to be used. Hit start - the program name will be typed followed by the mode the diagnostic is running in (uniprocessor or multi-processor) followed by the contents of the switch register and whether the Unibus Exerciser will be used to simulate power fails.

5.3 Notes And Warnings

1. Power failures in section 1 are only allowed when expected. Therefore in manual mode do not remove the power until the test number appears in the display register.
2. Power failures are not allowed during the execution of the End of Pass (EOP) routines. Note that the number displayed during the End of Pass is the Pass number NOT the test number.
3. When running the diagnostic in uniprocessor mode on a single CPU that is part of a multiprocessor system make sure that the IIST configuration switch for the respective CPU is in the STAND ALONE position and the IIST ENABLE switch is in the OFF LINE position.
4. When running in multiprocessor mode insure that the subsystem under test is isolated from any system that is still running by switching the CPU's under test to an alternate position with the IIST CONFIGURATION switches. For example, if the default system normally runs on the 'SYSTEM A' position, switch the subsystem under test to the 'SYSTEM B' position.

5.4 Uniprocessor Mode, No UBE

The diagnostic will instruct you to "interrupt the power after the test number appears on the display". Interrupt the power only at this time. If the test is successful then the next test number should appear in the display. When the End of Pass is reached the Pass count will be typed. There are no error reports in Section 1 (except for unexpected traps to 4 and 114.) Normally, an error results in a processor halt.

5.5 Uniprocessor Mode, With UBE

In this mode the UBE is used to perform the power Fail procedure. No "interrupt the power..." message is typed before the UBE takes action. The test number and pass number however do appear in the display register. An EOP message is typed equivalent to that of Section 5.4. Note that in this mode only the CPU logic involved in the power-fail sequence is tested. Failures that may occur because of problems in the Power system will go undetected.

5.6 Multiprocessor Mode, No UBE

In multiprocessor mode the CPU that is initially started becomes the 'MASTER' CPU. What this means is that all error messages and typeouts will appear on this CPU's console. This CPU is also responsible for startup of 'slave' CPU's. Slave CPU's are the remaining CPU's that are scheduled to participate in the test by their appropriate bit being set in switches 0-3 of the master CPU's SWR. When start is depressed and multi-processor mode is specified the program name, test mode, and switch register setting will be typed upon the master CPU console. The remaining CPU's will then be booted and then interrupted through the masters IIST. This will be followed by an "interrupt the power..." message.

The ID of the processor responsible for typeouts and error messages will precede the message:

- 0>
- 1>
- n>

Run each processor through section 1 one at a time. After power-failing the last test in Section 1 (Test 16) on one CPU go on to the next CPU, etc. After the last test on the last CPU has been completed in Section 1, the operator will receive instructions at console.

Whereas there are no prompts printed at the console in Section 1, all power fails in Section 2 must be done in exact agreement with the typed-out instructions.

Section 2 prompts the operator to remove the power from a particular element of the system. If the expected results occur then the next element is tried. When all relevant elements have been tried then the diagnostic precedes to the next test. Section 2 contains tests 17 through 25.

5.7 Multiprocessor Mode, with UBE

This mode is the same as 'multiprocessor mode, no UBE' except that Section 1 will be done without manual intervention. All CPUs that are to participate must have a UBE module.

6.0 ERRORS AND ERROR REPORTING

Error reports are always typed out on the console of the CPU that was first started. This is the 'master' CPU in multiprocessor mode. All error messages are preceded by a tag as follows 'n>' where n's the IIST self-ID of the CPU encountering the error. Typing in multiprocessor mode is always done by the master. If the master is without power when typing is required, the messages will be queued and printed when power has been restored.

7.0 TEST DESCRIPTIONS

The tests that are found in section I are simple power fail tests that guarantee that the proper machine states are entered on power fail and power up. The test names are self-explanatory.

Section 1

1. Simple Down/Up test (Kernal node)
2. Program Volatility Test
Verify that the memory bank containing the program will not be corrupted by CPU power fails.
3. Simple Down/Up test (Supervisor mode)
4. Simple Down/Up Test (User mode)
5. Power fail with odd address
6. Power Fail in the Red Zone
7. Power Fail with memory timeout (kernal)
10. Power Fail in the yellow zone.
11. Power Fail with resets.
12. Power Fail with odd address (Supervisor).
13. Power Fail with Timeout (Supervisor)
14. Power Fail with odd address (User)
15. Power Fail with timeout (User)
16. Memory Management Abort Test

After all CPUs reach the beginning of Section 2, each CPU sizes for RPO4/5/6 massbus devices. If no devices are found the following message is printed:

No Massbus Device Available On CPU #n

If the only massbus device found has its PGM bit set, the following message is printed:

Warning: Drive #n
On CPU #n
Is accessible over Ports A and B

and will be used later in this diagnostic.

17. Check 'BRK' & 'DCF' FLAGS during power fail. Insure that the IIST's of functioning CPU's receive BRK & DCF signals corresponding to the CPU that performed a Power Fail.

The operator will be prompted with messages of the following type:

Power Fail CPU #n

After restoring power, each of the other CPUs should report:

CPU Interrupt As Expected

20. Check power fail during high memory activity. Insure that power down sequences can be performed by a CPU while other CPU's are contending for the memory bus.

The operator will be prompted with messages of the following type:

Power Fail CPU #n

There is no report from the other CPUs after restoring power.

21. Check power fail during massbus transfer. Insure that power down sequences can be performed by a CPU while other CPU's are performing massbus read operations. Also verify that the read operations don't experience any loss of data due to the power condition of an uninvolved CPU.

The operator will be prompted with messages of the following type:

Power Fail CPU #n

L 1

If there are no massbus devices for the other CPUs to use, then the following messages are printed instead of the above message:

No Massbus Device Available On CPU #n Proceeding
To Next CPU.

There is no report from the other CPUs after restoring power.

22. Insure that a loss of AC power on a MKA11 semiconductor memory box does not cause a power fail sequence to occur on any processor that has a disabled part to that box.

The operator will be prompted with messages of the type:

Get Set To Power Fail Mem Box #n

Put battery backup on all memory boxes
Make all memory ports offline only on mem box to be power failed
Make all CPU power-up switches 'Run or Halt'

Now Power Fail The Mem Box
Restore power 5 seconds after power fail
Restore all memory ports online
Then type any character at the master console
No CPU should report a power fail

The master should report 'OK'.

23. Check AC power fail on memory box. Insure that a loss of AC power on a MKA11 semiconductor memory box causes a power fail sequence to occur on any processor that has an enabled port to that box.

The operator will be prompted with message of the type:

Get Set To Power Fail Mem Box #n

Put battery backup on all memory boxes
Make all memory ports online
Make all CPU power-up switches 'Run or Halt'

Now power fail the mem box

Restore power 5 seconds after power fail then type any character at the master console. Each CPU should report a power failure.

Each CPU should report the following message:

Power Failure On CPU As Expected

24. Check DC Power Loss on a memory box. Insure that the slave CPU(s) specified through the IIST Boot/control panel perform a boot operation when AC & DC power are restored to the memory box. M 1

The operator will be prompted with message of the type:

Get Set to Power Fail Mem Box #n

Disable battery backup on mem box to be power-failed. Put all slave CPU mem ports online. Make master CPU mem port offline only on box to be power-failed. Make all CPU power-up switches 'RUN OR Boot'.

Now Power Fail The Mem Box

Restore power 5 seconds after power fail
Restore all mem ports online
Restore all CPU power-up switches to 'Run or Halt'
Then type any character at the master console
Each slave should report an interrupt

Each slave CPU should report the following message:

CPU Interrupt As Expected

25. Check system recovery on power fail. Insure that a total momentary loss of AC power on a system level is recoverable without operator intervention.

The operator will be prompted with the following message:

Get Set to Power Fail Entire System...

Put battery backup on all mem boxes
Make all memory ports online
Make all CPU power-up switches 'Run or Halt'

Now Power Fail the Entire System
Restore power 5 seconds after power fail

Each CPU should report the following:

8.0 MISCELLANEOUS

SEQ 0013

Test 24 will not be done on the memory box with base address 0. Therefore, in order to fully test all system elements the operator should restart this diagnostic a second time and switch the box with base address 0 with another box (using the thumbwheels to switch base addresses). The operator should also switch master CPUs on the restart.

Test 22 will be skipped if there is only one MKA11 memory box. Power failing through Section 1 with the UBE will not necessarily test power fail during odd address trap, timeout, etc. (it depends on when the UBE starts the power down). Therefore, it is recommended to manually power fail Section 1!

a

57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112

```
::* IS NOT BEING USED, THE OPERATOR MUST POWER FAIL EACH CPU  
::* MANUALLY THRU THE 16 TESTS OF SECTION 1 TO GET THE CPU TO THE  
::* ENTRY POINT OF SECTION 2.  
::* BEFORE STARTING THE PROGRAM, BE SURE THAT THE CPU POWER-UP  
::* ACTION SWITCHES (ON THE IIST FRONT PANEL) ARE ALL  
::* 'RUN OF HALT', THE IIST ENABLE SWITCHES ARE ONLINE, AND  
::* THE IIST CONFIGURATION SWITCHES ARE EITHER ALL SYSTEM 0 OR 1.  
::* ALL PARTICIPATING MEMORY BOXES SHOULD BE ONLINE TO ALL PARTIC-  
::* IPATING CPUS.  
::*****
```

SWITCH REGISTER DEFINITIONS

```
::THE SWITCHES ARE DEFINED BY THE MASTER CPU ONLY,  
::WHERE THE MASTER IS THE CPU THAT STARTS OR RESTARTS  
::THE PROGRAM. EACH SLAVE CPU USES THE SWITCH DEFINITIONS  
::OF THE MASTER.
```

```
::THE PROGRAM WILL NOT INTERPRET CHANGES TO THE  
::SWITCHES IN MULTIPROCESSOR MODE WHILE THE PROGRAM IS RUNNING. THE  
::PROGRAM MUST BE RESTARTED.
```

```
::SW15=1 HALT ON ERROR  
  
::SW14=1 LOOP ON TEST (SECTION 1 ONLY)  
::SW08=1 ENABLE SYSTEM POWER FAIL TEST (TEST 25)  
::SW07=1 DISABLE SECTION 1 TESTS (MULTIPROCESSOR MODE ONLY)  
::SW06=1 ENABLE MULTIPROCESSOR MODE/SECTION 2 TESTS  
::SW05=1 ENABLE UNIBUS EXERCISERS (SECTION 1 ONLY, EACH CPU MUST HAVE A USE)  
::SW03=1 TEST CPU #3 (IIST SELF ID), MULTIPROCESSOR MODE ONLY  
::SW02=1 TEST CPU #2  
::SW01=1 TEST CPU #1  
::SW00=1 TEST CPU #0
```

SBTTL BASIC DEFINITIONS

```
::*INITIAL ADDRESS OF THE STACK POINTER  
STACK= 13776 ;;FIRST ADDRESS OF THE STACK  
KERSTK= STACK ;;KERNEL STACK  
SUPSTK= STACK-200 ;;SUPERVISOR STACK  
USESTK= STACK-300 ;;USER STACK  
.EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL  
.EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL  
PS= 177776 ;;PROCESSOR STATUS WORD  
.EQUIV PS,PSW  
STKLMT= 177774 ;;STACK LIMIT REGISTER  
PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER  
DSWR= 177570 ;;HARDWARE SWITCH REGISTER  
DDISP= 177570 ;;HARDWARE DISPLAY REGISTER  
LKS= 177546 ;;LINE CLOCK (KW11-L) STATUS REGISTER
```

MISCELLANEOUS DEFINITIONS

```
HT= 11 ;;CODE FOR HORIZONTAL TAB  
LF= 12 ;;CODE LINE FEED  
CR= 15 ;;CODE CARRIAGE RETURN  
CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
```

013776
013776
013576
013476

177776

177774
177772
177570
177570
177546

000011
000012
000015
000200

```
113      : *GENERAL PURPOSE REGISTER DEFINITIONS
114      000000 R0= %0      :: GENERAL REGISTER
115      000001 R1= %1      :: GENERAL REGISTER
116      000002 R2= %2      :: GENERAL REGISTER
117      000003 R3= %3      :: GENERAL REGISTER
118      000004 R4= %4      :: GENERAL REGISTER
119      000005 R5= %5      :: GENERAL REGISTER
120      000006 R6= %6      :: GENERAL REGISTER
121      000007 R7= %7      :: GENERAL REGISTER
122      .EQUIV R0,R10     :: GENERAL REGISTER
123      .EQUIV R1,R11     :: GENERAL REGISTER
124      .EQUIV R2,R12     :: GENERAL REGISTER
125      .EQUIV R3,R13     :: GENERAL REGISTER
126      .EQUIV R4,R14     :: GENERAL REGISTER
127      .EQUIV R5,R15     :: GENERAL REGISTER
128      000006 SP= %6     :: STACK POINTER
129      .EQUIV SP,KSP     :: KERNEL STACK POINTER
130      .EQUIV SP,SSP     :: SUPERVISOR STACK POINTER
131      .EQUIV SP,USP     :: USER STACK POINTER
132      000007 PC= %7     :: PROGRAM COUNTER
133
134      : *PRIORITY LEVEL DEFINITIONS
135      000000 PR0= 0      :: PRIORITY LEVEL 0
136      000040 PR1= 40     :: PRIORITY LEVEL 1
137      000100 PR2= 100    :: PRIORITY LEVEL 2
138      000140 PR3= 140    :: PRIORITY LEVEL 3
139      000200 PR4= 200    :: PRIORITY LEVEL 4
140      000240 PR5= 240    :: PRIORITY LEVEL 5
141      000300 PR6= 300    :: PRIORITY LEVEL 6
142      000340 PR7= 340    :: PRIORITY LEVEL 7
143
144      : *'SWITCH REGISTER' SWITCH DEFINITIONS
145      100000 SW15= 100000
146      040000 SW14= 40000
147      020000 SW13= 20000
148      010000 SW12= 10000
149      004000 SW11= 4000
150      002000 SW10= 2000
151      001000 SW09= 1000
152      000400 SW08= 400
153      000200 SW07= 200
154      000100 SW06= 100
155      000040 SW05= 40
156      000020 SW04= 20
157      000010 SW03= 10
158      000004 SW02= 4
159      000002 SW01= 2
160      000001 SW00= 1
161      .EQUIV SW09,SW9
162      .EQUIV SW08,SW8
163      .EQUIV SW07,SW7
164      .EQUIV SW06,SW6
165      .EQUIV SW05,SW5
166      .EQUIV SW04,SW4
167      .EQUIV SW03,SW3
168      .EQUIV SW02,SW2
```

```

169      .EQUIV SW01,SW1
170      .EQUIV SW00,SW0
171
172      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
173      100000      BIT15= 100000
174      040000      BIT14= 40000
175      020000      BIT13= 20000
176      010000      BIT12= 10000
177      004000      BIT11= 4000
178      002000      BIT10= 2000
179      001000      BIT09= 1000
180      000400      BIT08= 400
181      000200      BIT07= 200
182      000100      BIT06= 100
183      000040      BIT05= 40
184      000020      BIT04= 20
185      000010      BIT03= 10
186      000004      BIT02= 4
187      000002      BIT01= 2
188      000001      BIT00= 1
189      .EQUIV BIT09,BIT9
190      .EQUIV BIT08,BIT8
191      .EQUIV BIT07,BIT7
192      .EQUIV BIT06,BIT6
193      .EQUIV BIT05,BIT5
194      .EQUIV BIT04,BIT4
195      .EQUIV BIT03,BIT3
196      .EQUIV BIT02,BIT2
197      .EQUIV BIT01,BIT1
198      .EQUIV BIT00,BIT0
199
200      ;*BASIC 'CPU' TRAP VECTOR ADDRESSES
201      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
202      000010      RESVEC= 10         ;;RESERVED AND ILLEGAL INSTRUCTIONS
203      000014      TBITVEC=14         ;;'T' BIT
204      000014      TRTVEC= 14         ;;TRACE TRAP
205      000014      BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
206      000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
207      000024      PWRVEC= 24         ;;POWER FAIL
208      000030      EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
209      000034      TRAPVEC=34         ;;'TRAP' TRAP
210      000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
211      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
212      000100      LKVEC= 100         ;;LINE CLOCK (KW11-L) VECTOR
213      000114      CACHVEC=114        ;;CACHE ERROR INTERRUPT VECTOR
214      000240      PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
215      000250      MMVEC= 250         ;;MEMORY MANAGEMENT VECTOR
216      .SBTTL CACHE REGISTER DEFINITIONS
217
218
219      177740      LOADRS = 177740      ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
220      177742      HIADRS = 177742     ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
221      177744      MEMERR = 177744     ;;CACHE ERROR REGISTER
222      177746      CONTRL = 177746    ;;MEMORY CONTROL REGISTER
223      177750      MAINT = 177750      ;;MEMORY MAINTENANCE REGISTER
224      177752      HITMIS = 177752    ;;HIT MISS REGISTER '1' IMPLIES HIT IN CACHE
  
```

```

225
226      .SBTTL CPU REGISTER DEFINITIONS
227
228
229      177760      SIZELO = 177760      ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
230      ;;TO GET TO THE LAST 32 WORDS OF MEMORY
231      177762      SIZEHI = 177762      ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
232      ;;CURRENTLY ALL ZERO
233      177764      SYSTID = 177764      ;;SYSTEM ID REGISTER
234      177766      CPUERR = 177766      ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
235      ;;THE TRAP TO ERRVEC (000004)
236
237
238
  
```

.SBTTL MEMORY MANAGEMENT DEFINITIONS

```

239
240
241      ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
242
243
244      177572      MMR0= 177572
245      177574      MMR1= 177574
246      177576      MMR2= 177576
247      172516      MMR3= 172516
248      .EQUIV MMR0,SR0
249      .EQUIV MMR1,SR1
250      .EQUIV MMR2,SR2
251      .EQUIV MMR3,SR3
  
```

;*USER 'I' PAGE DESCRIPTOR REGISTERS

```

252
253
254
255      177600      UIPDR0= 177600
256      177602      UIPDR1= 177602
257      177604      UIPDR2= 177604
258      177606      UIPDR3= 177606
259      177610      UIPDR4= 177610
260      177612      UIPDR5= 177612
261      177614      UIPDR6= 177614
262      177616      UIPDR7= 177616
  
```

;*USER 'D' PAGE DESCRIPTOR REGISTORS

```

263
264
265
266      177620      UDPDR0= 177620
267      177622      UDPDR1= 177622
268      177624      UDPDR2= 177624
269      177626      UDPDR3= 177626
270      177630      UDPDR4= 177630
271      177632      UDPDR5= 177632
272      177634      UDPDR6= 177634
273      177636      UDPDR7= 177636
  
```

;*USER 'I' PAGE ADDRESS REGISTERS

```

274
275
276
277      177640      UIPAR0= 177640
278      177642      UIPAR1= 177642
279      177644      UIPAR2= 177644
280      177646      UIPAR3= 177646
  
```

281	177650	UIPAR4= 177650
282	177652	UIPAR5= 177652
283	177654	UIPAR6= 177654
284	177656	UIPAR7= 177656
285		
286		:*USER 'D' PAGE ADDRESS REGISTERS
287		
288	177660	UDPAR0= 177660
289	177662	UDPAR1= 177662
290	177664	UDPAR2= 177664
291	177666	UDPAR3= 177666
292	177670	UDPAR4= 177670
293	177672	UDPAR5= 177672
294	177674	UDPAR6= 177674
295	177676	UDPAR7= 177676
296		
297		:*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
298		
299	172200	SIPDR0= 172200
300	172202	SIPDR1= 172202
301	172204	SIPDR2= 172204
302	172206	SIPDR3= 172206
303	172210	SIPDR4= 172210
304	172212	SIPDR5= 172212
305	172214	SIPDR6= 172214
306	172216	SIPDR7= 172216
307		
308		:*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
309		
310	172220	SDPDR0= 172220
311	172222	SDPDR1= 172222
312	172224	SDPDR2= 172224
313	172226	SDPDR3= 172226
314	172230	SDPDR4= 172230
315	172232	SDPDR5= 172232
316	172234	SDPDR6= 172234
317	172236	SDPDR7= 172236
318		
319		:*SUPERVISOR 'I' PAGE ADDRESS REGISTERS
320		
321	172240	SIPAR0= 172240
322	172242	SIPAR1= 172242
323	172244	SIPAR2= 172244
324	172246	SIPAR3= 172246
325	172250	SIPAR4= 172250
326	172252	SIPAR5= 172252
327	172254	SIPAR6= 172254
328	172256	SIPAR7= 172256
329		
330		:*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
331		
332	172260	SDPAR0= 172260
333	172262	SDPAR1= 172262
334	172264	SDPAR2= 172264
335	172266	SDPAR3= 172266
336	172270	SDPAR4= 172270

337 172272 SDPAR5= 172272
338 172274 SDPAR6= 172274
339 172276 SDPAR7= 172276

;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS

340
341
342
343 172300 KIPDR0= 172300
344 172302 KIPDR1= 172302
345 172304 KIPDR2= 172304
346 172306 KIPDR3= 172306
347 172310 KIPDR4= 172310
348 172312 KIPDR5= 172312
349 172314 KIPDR6= 172314
350 172316 KIPDR7= 172316

;*KERNEL 'D' PAGE DESCRIPTOR REGISTERS

351
352
353
354 172320 KDPDR0= 172320
355 172322 KDPDR1= 172322
356 172324 KDPDR2= 172324
357 172326 KDPDR3= 172326
358 172330 KDPDR4= 172330
359 172332 KDPDR5= 172332
360 172334 KDPDR6= 172334
361 172336 KDPDR7= 172336

;*KERNEL 'I' PAGE ADDRESS REGISTERS

362
363
364
365 172340 KIPAR0= 172340
366 172342 KIPAR1= 172342
367 172344 KIPAR2= 172344
368 172346 KIPAR3= 172346
369 172350 KIPAR4= 172350
370 172352 KIPAR5= 172352
371 172354 KIPAR6= 172354
372 172356 KIPAR7= 172356

;*KERNEL 'D' PAGE ADDRESS REGISTERS

373
374
375
376 172360 KDPAR0= 172360
377 172362 KDPAR1= 172362
378 172364 KDPAR2= 172364
379 172366 KDPAR3= 172366
380 172370 KDPAR4= 172370
381 172372 KDPAR5= 172372
382 172374 KDPAR6= 172374
383 172376 KDPAR7= 172376

.SBTTL UNIBUS MAP REGISTER DEFINITIONS

384
385
386
387
388
389
390
391
392
;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'

UNIBUS MAP REGISTER DEFINITIONS

393		
394	170200	MAPL00 = 170200
395	170202	MAPH00 = 170202
396	170204	MAPL01 = 170204
397	170206	MAPH01 = 170206
398	170210	MAPL02 = 170210
399	170212	MAPH02 = 170212
400	170214	MAPL03 = 170214
401	170216	MAPH03 = 170216
402	170220	MAPL04 = 170220
403	170222	MAPH04 = 170222
404	170224	MAPL05 = 170224
405	170226	MAPH05 = 170226
406	170230	MAPL06 = 170230
407	170232	MAPH06 = 170232
408	170234	MAPL07 = 170234
409	170236	MAPH07 = 170236
410	170240	MAPL10 = 170240
411	170242	MAPH10 = 170242
412	170244	MAPL11 = 170244
413	170246	MAPH11 = 170246
414	170250	MAPL12 = 170250
415	170252	MAPH12 = 170252
416	170254	MAPL13 = 170254
417	170256	MAPH13 = 170256
418	170260	MAPL14 = 170260
419	170262	MAPH14 = 170262
420	170264	MAPL15 = 170264
421	170266	MAPH15 = 170266
422	170270	MAPL16 = 170270
423	170272	MAPH16 = 170272
424	170274	MAPL17 = 170274
425	170276	MAPH17 = 170276
426	170300	MAPL20 = 170300
427	170302	MAPH20 = 170302
428	170304	MAPL21 = 170304
429	170306	MAPH21 = 170306
430	170310	MAPL22 = 170310
431	170312	MAPH22 = 170312
432	170314	MAPL23 = 170314
433	170316	MAPH23 = 170316
434	170320	MAPL24 = 170320
435	170320	MAPH24 = 170320
436	170324	MAPL25 = 170324
437	170326	MAPH25 = 170326
438	170330	MAPL26 = 170330
439	170332	MAPH26 = 170332
440	170334	MAPL27 = 170334
441	170336	MAPH27 = 170336
442	170340	MAPL30 = 170340
443	170342	MAPH30 = 170342
444	170344	MAPL31 = 170344
445	170346	MAPH31 = 170346
446	170350	MAPL32 = 170350
447	170352	MAPH32 = 170352
448	170354	MAPL33 = 170354

449	70356	MAPH33 = 170356
450	170360	MAPL34 = 170360
451	170362	MAPH34 = 170362
452	170364	MAPL35 = 170364
453	170366	MAPH35 = 170366
454	170370	MAPL36 = 170370
455	170372	MAPH36 = 170372
456	170374	MAPL37 = 170374
457	170376	MAPH37 = 170376
458		.EQUIV MAPL00,MAPL0
459		.EQUIV MAPH00,MAPH0
460		.EQUIV MAPL01,MAPL1
461		.EQUIV MAPH01,MAPH1
462		.EQUIV MAPL02,MAPL2
463		.EQUIV MAPH02,MAPH2
464		.EQUIV MAPL03,MAPL3
465		.EQUIV MAPH03,MAPH3
466		.EQUIV MAPL04,MAPL4
467		.EQUIV MAPH04,MAPH4
468		.EQUIV MAPL05,MAPL5
469		.EQUIV MAPH05,MAPH5
470		.EQUIV MAPL06,MAPL6
471		.EQUIV MAPH06,MAPH6
472		.EQUIV MAPL07,MAPL7
473		.EQUIV MAPH07,MAPH7
474	170016	UBCR2=170016

.SBTTL IIST REGISTER DEFINITIONS

:: IIST INTERNAL REGISTERS

482	000000	PGTE = 0	:: PROGRAM-GENERATED TRANSMISSION ENABLE
483	000001	PGCS = 1	:: PROGRAM-GENERATED CONTROL STATUS
484	000002	STTE = 2	:: SANITY-TIMER TRANSMISSION ENABLE
485	000003	STCS = 3	:: SANITY-TIMER CONTROL STATUS
486	000004	IMSK = 4	:: INPUT MASK
487	000005	PGF = 5	:: PROGRAM GENERATED FLAGS
488	000006	STF = 6	:: SANITY-TIMER FLAGS
489	000007	DCF = 7	:: DCLO/DISCONNECT FLAGS
490	000010	EXC = 10	:: EXCEPTIONS
491	000015	MTC = 15	:: MAINTAINANCE CONTROL

:: IIST INTERRUPT VECTOR

.SBTTL CONSOLE SWITCH SETTINGS

497		:: WHEN THESE SWITCHES ARE ENABLED, IT SPECIFIES TO THE MASTER	
498		:: THAT THE CPU WITH THE IIST SELF ID CORRESPONDING TO THE	
499		:: EQUIVILENT BIT POSITION IS EXPECTED TO PARTICIPATE IN THIS TEST	
500			
501			
502	000001	CP0=BIT0	: CPU0 MASK LOCATION
503	000002	CP1=BIT1	: CPU1 MASK LOCATION
504	000004	CP2=BIT2	: CPU2 MASK LOCATION

```

505          000010          CP3=BIT3          ;CPU3 MASK LOCATION
506
507          ;: THE FOLLOWING SWITCHES CONTROL THE EXECUTION STREAM OF THE
508          ;: TEST.
509
510          000100          MPSW= BIT6          ;ENABLE FOR MULTIPROCESSOR CONFIGURATIONS
511          000040          UBESW= BITS          ;ENABLE IF UNIBUS EXERCISER IS TO BE USED
512          001000          LOE= BIT9          ;ENABLE FOR LOOPING ON ERRORS
513          040000          LOT= BIT14         ;ENABLE FOR LOOPING ON TEST
514          100000          HOE= BIT15         ;HALT ON ERROR
515
516
517
518
  
```

.SBTTL RJPO4 DEVICE REGISTERS

```

519          176700          RPCS1=176700        ;: CONTROL AND STATUS 1
520          176702          RPWC =176702        ;: WORD COUNT REGISTER
521          176704          RPBA =176704        ;: UNIBUS ADDRESS
522          176706          RPDA =176706        ;: DESIRED SECTOR/TRACK ADDRESS
523          176710          RPCS2=176710        ;: CONTROL AND STATUS 2
524          176712          RPDS =176712        ;: DRIVE STATUS
525          176714          RPER1=176714        ;: ERROR REGISTER 1
526          176716          RPAS =176716        ;: ATTENTION SUMMARY
527          176720          RPLA =176720        ;: LOOK-AHEAD REGISTER
528          176722          RPDB =176722        ;: DATA BUFFER REGISTER
529          176724          RPMR =176724        ;: MAINTENANCE REGISTER
530          176726          RPDT =176726        ;: DRIVE TYPE
531          176730          RPSN =176730        ;: SERIAL NUMBER REGISTER
532          176732          RPOF =176732        ;: OFFSET REGISTER
533          176734          RPDC =176734        ;: DESIRED CYLINDER REGISTER
534          176736          RPCC =176736        ;: CURRENT CYLINDER REGISTER
535          176740          RPER2=176740        ;: ERROR REGISTER 2
536          176742          RPER3=176742        ;: ERROR REGISTER 3
537          176744          RPEC1=176744        ;: ECC POSITION REGISTER
538          176746          RPEC2=176746        ;: ECC PATTERN REGISTER
  
```

```

539      .SBTTL POWER FAIL FUNCTION TABLE BIT DEFINITIONS
540
541      010000      NCX=BIT12      ;;DON'T SAVE MM REGISTERS
542      004000      TI =BIT11      ;;TIME THE POWER FAIL
543      002000      NS =BIT10      ;;DON'T PERFORM A REGISTER SAVE
544      001000      SID=BIT9       ;;SEND ERROR ON ILLEGAL DOWN
545      000400      SIU=BIT8       ;;SEND ERROR ON ILLEGAL UP
546      000200      SED=BIT7       ;;SEND ERROR ON DOWN
547      000100      SEU=BIT6       ;;SEND ERROR ON UP
548      000040      SSD=BIT5       ;;SEND SIGNAL ON DOWN
549      000020      SSU=BIT4       ;;SEND SIGNAL ON UP.
550
551      .SBTTL TRAP CATCHER
552
553      000000      .=0
554      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
555      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
556      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
557      000174      000174      .=174
558      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
559      000176      000000      SWREG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
560
561      .SBTTL ACT11 HOOKS
562
563      ;*****
564      ;HOOKS REQUIRED BY ACT11
565      000200      $SVPC=.      ;SAVE PC
566      000046      .=46
567      000046      036112      $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
568      000052      000000      .=52
569      000052      000000      .WORD 0      ;;2)SET LOC.52 TO ZERO
570      000200      .=$$VPC      ;; RESTORE PC
571
572
573      .SBTTL LOAD START AND RESTART VECTORS
574      000200      .=200
575      000200      000137      020064      JMP STRT      ;LOAD 200 WITH A JUMP TO START OF TEST
576      000220      000220
577      000220      004737      020000      JSR PC, RESTRT      ;LOAD 220 WITH A JUMP TO THE RESTART CODE
578      000224      000137      020064      JMP STRT
579
580
581
582
583
584      .SBTTL APT PARAMETER BLOCK
585
586      ;*****
587      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
588      ;*****
589      000230      000230      .$X=.      ;;SAVE CURRENT LOCATION
590      000024      000024      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
591      000024      000200      200      ;;FOR APT START UP
592      000044      000044      .=44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
593      000044      000230      $APTHDR ;;POINT TO APT HEADER BLOCK
594      000230      000230      . .$X      ;;RESET LOCATION COUNTER
  
```

595
596
597
598
599 000230
600 000230 000000
601 000232 014336
602 000234 000000
603 000236 000000
604 000240 000000
605 000242 000052

: SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
: INTERFACE SPEC.
\$APTHD:
\$HIBTS: .WORD 0 ;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MADR: .WORD \$MAIL ;; ADDRESS OF APT MAILBOX (BITS 0-15)
\$STMT: .WORD ;; RUN TIM OF LONGEST TEST
\$PASTM: .WORD ;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD ;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
\$ETEND-\$MAIL/2 ;; LENGTH MAILBOX-ETABLE (WORDS)


```
606 .SBTTL COMMON TAGS
607
608 ;:*****
609 ;:THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
610 ;:USED IN THE PROGRAM.
611
612 014000 014000 $CMTAG: .=14000 ;:START OF COMMON TAGS
613 014000 000000 $STSTM: .WORD 0 ;:CONTAINS THE TEST NUMBER
614 014000 000000 .BYTE 0
615 014002 000000 .BYTE 0
616 014003 000000 .BYTE 0
617 014004 000000 .BYTE 0
618 014005 000000 .BYTE 0
619 014006 000000 $ERFLG: .BYTE 0 ;:CONTAINS ERROR FLAG
620 014007 000000 .BYTE 0
621 014010 000000 .BYTE 0
622 014011 000000 .BYTF 0
623 014012 000000 $ICNT: .WORD 0 ;:CONTAINS SUBTEST ITERATION COUNT
624 014014 000000 .WORD 0
625 014016 000000 .WORD 0
626 014020 000000 .WORD 0
627 014022 000000 $LPADR: .WORD 0 ;:CONTAINS SCOPE LOOP ADDRESS
628 014024 000000 .WORD 0
629 014026 000000 .WORD 0
630 014030 000000 $LPERR: .WORD 0 ;:CONTAINS SCOPE RETURN FOR ERRORS
631 014032 000000 .WORD 0
632 014034 000000 .WORD 0
633 014036 000000 .WORD 0
634 014040 000000 .WORD 0
635 014042 000000 $ERTTL: .WORD 0 ;:CONTAINS TOTAL ERRORS DETECTED
636 014044 000000 .WORD 0
637 014046 000000 .WORD 0
638 014050 000000 .WORD 0
639 014052 000000 $ITEMB: .BYTE 0 ;:CONTAINS ITEM CONTROL BYTE
640 014054 000000 .BYTE 0
641 014055 000000 .BYTE 0
642 014056 000000 .BYTE 0
643 014057 000000 .BYTE 0
644 014060 001000 $ERMAX: .BYTE 1 ;:CONTAINS MAX. ERRORS PER TEST
645 014061 001000 .BYTE 1
646 014062 001000 .BYTE 1
647 014063 001000 .BYTE 1
648 014064 000000 $ERRPC: .WORD 0 ;:CONTAINS PC OF LAST ERROR INSTRUCTION
649 014066 000000 .WORD 0
650 014070 000000 .WORD 0
651 014072 000000 .WORD 0
652 014074 000000 $ERRSP: .WORD 0 ;:CONTAINS SP OF CPU IN ERROR
653 014076 000000 .WORD 0
654 014100 000000 .WORD 0
655 014102 000000 .WORD 0
656 014104 000000 $GDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'GOOD' DATA
657 014106 000000 .WORD 0
658 014110 000000 .WORD 0
659 014112 000000 .WORD 0
660 014114 000000 $BDADR: .WORD 0 ;:CONTAINS ADDRESS OF 'BAD' DATA
661 014116 000000 .WORD 0
```

662	014120	000000				.WORD	0	
663	014122	000000				.WORD	0	
664	014124	000000			\$GDDAT:	.WORD	0	::CONTAINS 'GOOD' DATA
665	014126	000000				.WORD	0	
666	014130	000000				.WORD	0	
667	014132	000000				.WORD	0	
668	014134	000			\$EOPSG:	.BYTE	0	::THIS TABLE HOLDS THE END OF PASS
669	014135	000				.BYTE	0	
670	014136	000				.BYTE	0	
671	014137	000				.BYTE	0	
672								
673	014140	000000			\$BDDAT:	.WORD	0	::CONTAINS 'BAD' DATA
674	014142	000000				.WORD	0	
675	014144	000000				.WORD	0	
676	014146	000000				.WORD	0	
677	014150	000000				.WORD	0	::RESERVED--NOT TO BE USED
678	014152	000000				.WORD	0	
679	014154	000			\$AUTOB:	.BYTE	0	::AUTOMATIC MODE INDICATOR
680	014155	000			\$INTAG:	.BYTE	0	::INTERRUPT MODE INDICATOR
681	014156	000000				.WORD	0	
682	014160	177570			\$WR:	.WORD	DSWR	::ADDRESS OF SWITCH REGISTER
683	014162	177570				.WORD	DSWR	
684	014164	177570				.WORD	DSWR	
685	014166	177570				.WORD	DSWR	
686	014170	177570			DISPLAY:	.WORD	DDISP	::ADDRESS OF DISPLAY REGISTER
687	014172	177570				.WORD	DDISP	
688	014174	177570				.WORD	DDISP	
689	014176	177570				.WORD	DDISP	
690	014200	013776			\$SSTP:	.WORD	STACK	::STACK INITIALIZATION FOR CPU0
691	014202	011776				.WORD	STACK-2000	::CPU1
692	014204	007776				.WORD	STACK-4000	::CPU2
693	014206	003776				.WORD	STACK-10000	::CPU3
694	014210	177560			\$TKS:	177560		::TTY KBD STATUS
695	014212	177562			\$TKB:	177562		::TTY KBD BUFFER
696	014214	177564			\$TPS:	177564		::TTY PRINTER STATUS REG. ADDRESS
697	014216	177566			\$TPB:	177566		::TTY PRINTER BUFFER REG. ADDRESS
698	014220	000			\$NULL:	.BYTE	0	::CONTAINS NULL CHARACTER FOR FILLS
699	014221	002			\$FILLS:	.BYTE	2	::CONTAINS # OF FILLER CHARACTERS REQUIRED
700	014222	012			\$FILLC:	.BYTE	12	::INSERT FILL CHARS. AFTER A 'LINE FEED'
701	014223	000			\$TPFLG:	.BYTE	0	::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
702	014224	000000			\$ERGBL:	.WORD	0	
703	014226	177777			\$CPUID:	.WORD	-1	::THIS TABLE HOLDS THE PHYSICAL ID OF
704	014230	177777				.WORD	-1	::THE PARTICIPATING PROCESSORS ARRANGED
705	014232	177777				.WORD	-1	::IN LOGICAL ORDER.
706	014234	177777				.WORD	-1	
707	014236	000000			\$REGAD:	.WORD	0	::CONTAINS THE ADDRESS FROM
708								::WHICH (\$REGO) WAS OBTAINED
709	014240	000000	000000	000000	\$REG0:	.WORD	0,0,0,0	::CONTAINS ((\$REGAD)+0+6)
710	014246	000000						
711	014250	000000	000000	000000	\$REG1:	.WORD	0,0,0,0	::CONTAINS ((\$REGAD)+2+6)
712	014256	000000						
713	014260	000000	000000	000000	\$REG2:	.WORD	0,0,0,0	::CONTAINS ((\$REGAD)+4+6)
714	014266	000000						
715	014270	000000	000000	000000	\$REG3:	.WORD	0,0,0,0	::CONTAINS ((\$REGAD)+6+6)
716	014276	000000						
717	014300	000000	000000	000000	\$REG4:	.WORD	0,0,0,0	::CONTAINS ((\$REGAD)+10+6)

```
718 014306 000000
719 014310 000000
720 014312 000000
721 014314 000000
722 014316 000000
723 014320 000000
724 014322 000000
725 014324 000000
726 014326 000000
727 014330 000000
728 014332 077
729 014333 015
730 014334 000012
731
732
733
734
735
736 014336
737 014336 000000
738 014340 000000
739 014342 000000
740 014344 000000
741 014346 000000
742 014350 000000
743 014352 000000
744 014354 000000
745 014356
746 014356 000
747 014357 000
748 014360 000000
749 014362 000000
750 014364 000000
751
752
753
754
755
756
757 014366 000
758 014367 000
759
760
761
762
763 014370 000000
764
765 014372 000
766 014373 000
767 014374 000000
768 014376 000
769 014377 000
770 014400 000000
771 014402 000
772 014403 000
773 014404 000000

$TMP0: .WORD 0 ;;USER DEFINED
$TMP1: .WORD 0 ;;USER DEFINED
$TMP2: .WORD 0 ;;USER DEFINED
$TMP3: .WORD 0 ;;USER DEFINED
$TMP4: .WORD 0 ;;USER DEFINED
$ESCAPE:0 ;;ESCAPE ON ERROR ADDRESS
      .WORD 0
      .WORD 0
      .WORD 0
$QUES: .ASCII /?/ ;;QUESTION MARK
$CRLF: .ASCII <15> ;;CARRIAGE RETURN
$LF: .ASCII <12> ;;LINE FEED
:*****
.SBTTL APT MAILBOX-ETABLE
:*****
.EVEN
$MAIL: ;;APT MAILBOX
$MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
$FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
$TESTN: .WORD ATESTN ;;TEST NUMBER
$PASS: .WORD APASS ;;PASS COUNT
$DEVCT: .WORD ADEVCT ;;DEVICE COUNT
$UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
$MSGAD: .WORD AMSGAD ;;MESSAGE ADDRESS
$MSGGLG: .WORD AMSGLG ;;MESSAGE LENGTH
$ETABLE: ;;APT ENVIRONMENT TABLE
$ENV: .BYTE AENV ;;ENVIRONMENT BYTE
$ENVM: .BYTE AENVM ;;ENVIRONMENT MODE BITS
$SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
$USWR: .WORD AUSWR ;;USER SWITCHES
$CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
: *
: * BIT 15-11=CPU TYPE
: * 11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
: * 11/70=06,PDQ=07,Q=10
: * BIT 10=REAL TIME CLOCK
: * BIT 9=FLOATING POINT PROCESSOR
: * BIT 8=MEMORY MANAGEMENT
$MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
$MTYP1: .BYTE AMTYP1 ;;MEM. TYPE,BLK#1
: *
: * MEM.TYPE BYTE -- (HIGH BYTE)
: * 900 NSEC CORE=001
: * 300 NSEC BIPOLAR=002
: * 500 NSEC MOS=003
$MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
: * MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF 'TYPE' ABOVE
$MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
$MTYP2: .BYTE AMTYP2 ;;MEM.TYPE,BLK#2
$MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
$MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
$MTYP3: .BYTE AMTYP3 ;;MEM.TYPE,BLK#3
$MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
$MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
$MTYP4: .BYTE AMTYP4 ;;MEM.TYPE,BLK#4
$MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
```

```

774 014406 000000 $VECT1: .WORD AVECT1 ;; INTERRUPT VECTOR#1,BUS PRIORITY#1
775 014410 000000 $VECT2: .WORD AVECT2 ;; INTERRUPT VECTOR#2BUS PRIORITY#2
776 014412 000000 $BASE: .WORD ABASE ;; BASE ADDRESS OF EQUIPMENT UNDER TEST
777 014414 000000 $DEVM: .WORD ADEVM ;; DEVICE MAP
778 014416 000000 $CDW1: .WORD ACDW1 ;; CONTROLLER DESCRIPTION WORD#1
779 014420 000000 $CDW2: .WORD ACDW2 ;; CONTROLLER DESCRIPTION WORD#2
780 014422 000000 $DDW0: .WORD ADDW0 ;; DEVICE DESCRIPTOR WORD#0
781 014424 000000 $DDW1: .WORD ADDW1 ;; DEVICE DESCRIPTOR WORD#1
782 014426 000000 $DDW2: .WORD ADDW2 ;; DEVICE DESCRIPTOR WORD#2
783 014430 000000 $DDW3: .WORD ADDW3 ;; DEVICE DESCRIPTOR WORD#3
784 014432 000000 $DDW4: .WORD ADDW4 ;; DEVICE DESCRIPTOR WORD#4
785 014434 000000 $DDW5: .WORD ADDW5 ;; DEVICE DESCRIPTOR WORD#5
786 014436 000000 $DDW6: .WORD ADDW6 ;; DEVICE DESCRIPTOR WORD#6
787 014440 000000 $DDW7: .WORD ADDW7 ;; DEVICE DESCRIPTOR WORD#7
788 014442 000000 $DDW8: .WORD ADDW8 ;; DEVICE DESCRIPTOR WORD#8
789 014444 000000 $DDW9: .WORD ADDW9 ;; DEVICE DESCRIPTOR WORD#9
790 014446 000000 $DDW10: .WORD ADDW10 ;; DEVICE DESCRIPTOR WORD#10
791 014450 000000 $DDW11: .WORD ADDW11 ;; DEVICE DESCRIPTOR WORD#11
792 014452 000000 $DDW12: .WORD ADDW12 ;; DEVICE DESCRIPTOR WORD#12
793 014454 000000 $DDW13: .WORD ADDW13 ;; DEVICE DESCRIPTOR WORD#13
794 014456 000000 $DDW14: .WORD ADDW14 ;; DEVICE DESCRIPTOR WORD#14
795 014460 000000 $DDW15: .WORD ADDW15 ;; DEVICE DESCRIPTOR WORD#15

```

```

796
797
798 014462 $ETEND:
799

```

```

800 014462 000000 000000 000000 STOP: 0,0,0,0,0,0,0,0 ;MEM BOX UPPER BOUND ADDRESS TABLE
801 014470 000000 000000 000000

```

```

802 014476 000000 000000
803 014502 000000 000000 000000 START: 0,0,0,0,0,0,0,0 ;MEM BOX STARTING ADDRESS TABLE
804 014510 000000 000000 000000

```

```

805 014516 000000 000000
806 014522 000000 PWRFL: 0 ;=0 DON'T EXPECT CPU POWER FAIL,=1 EXPECT IT
807 014524 000000 YYY: 0 ;ERROR ROUTINE WORK LOC
808 014526 000000 BOOT: 0 ;=0 DON'T EXPECT CPU BOOT, -1 EXPECT IT
809 014530 000000 PATCHK: 0 ;=0 WRITE AND CHECK MEM PATTERN,=1 CHECK ONLY
810 014532 000000 HICORE: 0 ;=1 TURN ON MM ON POWER-UP
811 014534 000000 RELOUP: 0 ;=0 DON'T RELOCATE,=1 RELOCATE
812 014536 000000 RELODN: 0 ;=0 DON'T RELOCATE,=1 RELOCATE
813 014540 000000 EXIT: 0 ;=0 DON'T EXIT, =1 EXIT TEST
814 014542 000000 ENTR22: 0 ;CONTROL ENTRY INTO TEST 22
815 014544 000000 ENTR23: 0 ;CONTROL ENTRY INTO TEST 23
816 014546 000000 ENTR24: 0 ;CONTROL ENTRY INTO TEST 24
817 014550 000000 HIBOX: 0 ;RELOCATE TO THIS MEM BOX
818 014552 000000 000000 000000 SAVRG: 0,0,0,0 ;A PLACE TO SAVE A REGISTER

```

```

819 014560 000000
820 014562 000000 000000 000000 ROUTE: 0,0,0,0 ;TYPE TRAP ROUTE TABLE
821 014570 000000

```

```

822
823 014572 000000 000000 000000 SAV6: 0,0,0,0 ;SOME PLACE TO PUT THE SP
824 014600 000000

```

```

825 014602 000000 000000 000000 FLAG: 0,0,0,0 ;INSTRUCTION DOWN FLAG
826 014610 000000

```

```

827 014612 000000 000000 000000 PFFT: .WORD 0,0,0,0 ;;POWER FAIL FUNCTION TABLE
828 014620 000000

```

```

829 014622 000000 000000 000000 PFDT: .WORD 0,0,0,0 ;;POWER FAIL DURATION TABLE

```


MAINDEC-11-CEKBG-C
CEKBG.P11 13-MAR-80

PDP-11/70 SYSTEM POWER FAIL
APT MAILBOX-ETABLE

MACY11 30A(1052) 13-MAR-80 16:04 PAGE 19

SEQ 0031

886	017040	000400			BMSK: .WORD 400	:LIST INITIAL BOOT MASK	
887	017042	051266	061266	071266	BFADR: .WORD	DSKBUF, DSKBUF+10000, DSKBUF+20000, DSKBUF+30000	; MASSBUS TRANSFER BUF AR
888	017050	101266					
889	017052	000400	000400	000400	YELLIM: .WORD	400,400,400,400	; YELLOW ZONE BOUNDARY
890	017060	000400					
891	017062	177777			PUT: -1	:CPU UNDER TEST	
892	017064	177777			TYPLCK: -1	:ERROR/TYPE SEMAPHORE	
893	017066	006405			FACTOR: 2000./6	:POWER DWN FACTER	
894					.EVEN		

895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950

017070
017070 045274
017072 046563
017074 047166
017076 000000

017100 045336
017102 000000
017104 000000
017106 000000

017110 045404
017112 000000
017114 000000
017116 000000

017120 045451
017122 000000
017124 000000
017126 000000

017130 045505
017132 046461
017134 047144
017136 000000

017140 045534
017142 046461
017144 047144
017146 000000

017150 045564
017152 046514
017154 047154
017156 000000

.SBTTL ERROR POINTER TABLE

.*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
.*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
.*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
.*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
.*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

.* EM ::POINTS TO THE ERROR MESSAGE
.* DH ::POINTS TO THE DATA HEADER
.* DT ::POINTS TO THE DATA
.* D= ::POINTS TO THE DATA FORMAT

\$ERRTB:

:::ITEM 1
EM1 :UNEXPECTED POWER FAILURE ON CPU
DH10 :TESTNO ERRORPC
DT10 :\$STNM,\$ERRPC
0

:::ITEM 2
EM2 :UNEXPECTED POWER UP SEQUENCE ON CPU
0
0
0

:::ITEM 3
EM3 :ILLEGAL POWER DOWN SEQUENCE
0
0
0

:::ITEM 4
EM4 :ILLEGAL POWER UP SEQUENCE
0
0
0

:::ITEM 5
EM5 :UNEXPECTED TRAP TO LOCATION 4
DH5 :PID ERRORPC CPUERR
DT5 :\$REG0,\$ERRPC,\$REG1
0

:::ITEM 6
EM6 :UNEXPECTED TRAP TO 10
DH5 :PID ERRORPC CPUERR
DT5
0

:::ITEM 7
EM7 :UNEXPECTED TRAP TO 114
DH7 :PID ERRORPC CPUERR MEMERR
DT7 :\$REG0,\$ERRPC,\$REG1,\$REG2
0

951					
952			:::ITEM 10		
953	017160	045615	EM10	:	ADDRESS ON THE STACK IS WRONG
954	017162	046563	DH10	:	TESTNO ERRORPC
955	017164	047166	DT10	:	\$STNM,ERRORPC
956	017166	000000	0		
957					
958			:::ITEM 11		
959	017170	045651	EM11	:	OLD PS IS WRONG
960	017172	046606	DH11	:	TESTNO ERRORPC PS
961	017174	047174	DT11	:	\$STNM,\$ERRPC,\$REGO
962	017176	000000	0		
963					
964			:::ITEM 12		
965	017200	045673	EM12	:	ODD ADDRESS TRAP FAILED
966	017202	046563	DH10		
967	017204	047166	DT10		
968	017206	000000	0		
969					
970			:::ITEM 13		
971	017210	045725	EM13	:	MEMORY CORRUPTED ON POWER FAIL
972	017212	046640	DH12		
973	017214	047204	DT12		
974	017216	000000	0		
975					
976			:::ITEM 14		
977	017220	045766	EM14	:	TIMEOUT TRAP FAILED
978	017222	046727	DH14	:	TESTNO ERRORPC CPUERR
979	017224	047174	DT11	:	\$STNM,\$ERRPC,\$REGO
980	017226	000000	0		
981					
982			:::ITEM 15		
983	017230	046014	EM15	:	POWER FAIL RETURNED TO SOON
984	017232	046563	DH10		
985	017234	047166	DT10		
986	017236	000000	0		
987					
988			:::ITEM 16		
989	017240	046053	EM16	:	NOT ENOUGH OR TOO MABY INSTRUCTIONS EXECUTED
990	017242	046563	DH10		
991	017244	047166	DT10		
992	017246	000000	0		
993					
994			:::ITEM 17		
995	017250	046132	EM17	:	NO MEM. MANG. VIOLATION OR TRAP TO 4
996	017252	046727	DH14		
997	017254	047174	DT11		
998	017256	000000	0		
999					
1000			:::ITEM 20		
1001	017260	046201	EM20	:	NO IIST INTERRUPT
1002	017262	046762	DH20	:	TESTNO ISTID ACR PGTE PGCS
1003	017264	047214	DT20	:	\$STNM,\$REGO,\$REG1,\$REG2,\$REG3
1004	017266	000000	0		
1005					
1006			:::ITEM 21		

1007	017270	046225	EM21	: INCORRECT BRK AND/OR DCF FLAGS
1008	017272	047037	DH21	: TESTNO ISTID FOUND SHOULD BE
1009	017274	047230	DT21	: \$STNM, \$REG0, \$REG1, \$REG2
1010	017276	000000	0	
1011				
1012			:: ITEM 22	
1013	017300	046266	EM22	: CPU DID NOT TRAP TO 24
1014	017302	047107	DH22	: TESTNO ISTID ERRORPC
1015	017304	047242	DT22	: \$STNM, \$REG0, \$ERRPC
1016	017306	000000	0	
1017				
1018			:: ITEM 23	
1019	017310	046327	EM23	: CHECKSUM ON MASSBUS TRANSFER IS WRONG
1020	017312	047107	DH22	
1021	017314	047242	DT22	
1022	017316	000000	0	
1023				
1024			:: ITEM 24	
1025	017320	046377	EM24	: NO POWER FAIL ON CPU
1026	017322	046563	DH10	: TESTNO ERRORPC
1027	017324	047166	DT10	: \$STNM, \$ERRPC
1028	017326	000000	0	
1029				
1030			:: ITEM 25	
1031	017330	046426	EM25	: UNEXPECTED CPU INTERRUPT
1032	017332	046563	DH10	: TESTNO ERRORPC
1033	017334	047166	DT10	: \$STNM, \$ERRPC
1034	017336	000000	0	
1035				
1036				
1037				
1038				
1039				
1040				
1041		020000		

```

1042 020000 020000          RESTRT:  .=20000          ;;LOAD CODE ABOVE THE FIRST 4K (WORDS)
1043 020000 012704 016774  MOV    #CPUACT, R4      ;RESET THE VARIABLES
1044 020004 012724 000001  1$:   MOV    #1, (R4)+
1045 020010 022704 017026  CMP    #S2L1, R4
1046 020014 103373          BHS   1$
1047 020016 012704 014462  MOV    #STOP, R4
1048 020022 005024          CLR   (R4)+
1049 020024 022704 016772  2$:   CMP    #EXTRA, R4
1050 020030 103374          BHS   2$
1051 020032 012737 177777 017036  MOV    #-1, FIRST
1052 020040 012737 000400 017040  MOV    #400, BMSK
1053 020046 012737 177777 017062  MOV    #-1, PUT
1054 020054 012737 177777 017064  MOV    #-1, TYPLCK
1055 020062 000207          RI    PC          ;RETURN
1056
1057 020064 013706 014200  STRT:  MOV    $$STP, SP      ;INITIALIZE THE STACK
1058 020070 012704 014226  MOV    #CPUID, R4        ;;GET ADDRESS OF CPUID TABLE
1059 020074 012703 000004  MOV    #4, R3           ;;INIT COUNTER
1060 020100 012724 177777  4$:   MOV    #-1, (R4)+      ;;INIT TABLE WITH -1'S
1061 020104 077303          SOB   R3, 4$
1062          ;;INITIALIZE A FEW VECTORS
    
```

1063	020106	012737	036432	000030	MOV	#\$ERROR,@#EMTVEC	::EMT VECTOR FOR ERROR ROUTINE
1064	020114	005037	000032		CLR	@#EMTVEC+2	::LEVEL 0
1065	020120	012737	040520	000034	MOV	#\$TRAP,@#TRAPVEC	::TRAP VECTOR FOR TRAP CALLS
1066	020126	005037	000036		CLR	@#TRAPVEC+2	::LEVEL 0
1067	020132	012737	040606	000024	MOV	#PWRDIS,@#PWRVEC	::POINT TO POWER FAIL DISPATCH ROUTINE
1068	020140	005037	000026		CLR	@#PWRVEC+2	::LEVEL 0
1069	020144	012737	020330	000004	MOV	#25\$,@#ERRVEC	::SET UP CPU ERROR VECTOR INCASE SWR IS'NT THERE
1070	020152	012737	014722	014716	MOV	#TYPQUE+4,TYPQUE	::INITIALIZE REAR POINTER
1071	020160	012737	014722	014720	MOV	#TYPQUE+4,TYPQUE+2	::INITIALIZE FORWARD POINTER
1072	020166	005000			CLR	R0	::SET ID IN DISPLACEMENT REGS.
1073	020170	005001			CLR	R1	
1074	020172	023737	000042	000046	CMP	@#42, @#46	:UNDER ACT AUTO MODE?
1075	020200	001402			BEQ	40\$:BRANCH IF YES
1076	020202	104401	040644		TYPE	.TM1	:TYPE PROGRAM NAME
1077	020206			40\$:			
1078	020206	012737	177570	014160	MOV	#177570, SWR	::SET SWR
1079	020214	132737	000200	014357	BITB	#APTSIZE, \$ENVM	::SIZE UNDER APT?
1080	020222	001403			BEQ	50\$::BRANCH IF NO
1081	020224	012737	014360	014160	MOV	#SSWREG, SWR	::USE APT SWITCH REG
1082	020232	032777	000100	173720	BIT	#MPSW,@SWR	::IS HARDWARE SWITCH REG. THERE? AND MP SET?
1083	020240	001021			BNE	20\$::SWR IS THERE AND MP IS SET.
1084	020242	001435			BEQ	30\$::SWR IS THERE BUT MP IS NOT SET.
1085	020244	012737	040624	000004	MOV	#ERRDIS,@#ERRVEC	::RESET CPU ERROR VECTOR
1086	020252	012760	040630	014702	MOV	#CPUER,ERRTAB(R0)	::FLAG ALL UNEXPECTED TRAPS TO 4
1087	020260	012737	014360	014160	MOV	#SSWREG,SWR	::SETUP FOR SOFTWARE SWITCH REG
1088	020266	012737	000174	014170	MOV	#DISPREG,DISPLAY	
1089	020274	032777	000100	173656	BIT	#MPSW,@SWR	::IS MP SWITCH SET IN SOFTWARE SWITCH REG?
1090	020302	001415			BEQ	30\$::NOPE.
1091	020304	012737	040624	000004	MOV	#ERRDIS,@#ERRVEC	::RESET ERROR VECTOR
1092	020312	012760	040630	014702	MOV	#CPUER,ERRTAB(R0)	::FLAG ALL UNEXPECTED TRAPS TO 4
1093	020320	152737	000001	016737	BISB	#BIT0,MPF	::SET THE MP FLAG.
1094	020326	000416			BR	31\$::LET'S GO
1095	020330	062716	000004	25\$:	ADD	#4,(SP)	::SKIP RETURN
1096	020334	000002			RTI		::RETURN FROM TRAP
1097	020336	012737	040624	000004	MOV	#ERRDIS,@#ERRVEC	::RESET ERROR VECTOR
1098	020344	012760	040630	014702	MOV	#CPUER,ERRTAB(R0)	::FLAG ALL UNEXPECTED TRAPS TO 4
1099	020352	105037	016737		CLRB	MPF	::CLEAR THE MP FLAG
1100	020356	104401	041100		TYPE	.TM7	::[UNIPROCESSOR MODE IS IN EFFECT]
1101	020362	000425			BR	42\$::ENTER INTO NON MP EXECUTION STREAM
1102	020364	104401	041033	31\$:	TYPE	.TM6	::[MULTIPROCESSOR MODE IS IN EFFECT]
1103	020370	017737	173564	014660	MOV	@SWR,\$PSWR	::SET UP PSEUDO SWITCH REGISTER.
1104	020376	012704	014160		MOV	#SWR,R4	::POINT TO SWR TABLE
1105	020402	012705	000004		MOV	#4,R5	::SET COUNTER
1106	020406	012724	014660	41\$:	MOV	#SPSW,(R4)+	::LOAD THE SLAVE SWITCH REG. POINTERS
1107	020412	077503			SOB	R5,41\$::LOOP TILL DONE
1108	020414	104401	040676		TYPE	.TM2	::'SWITCH REGISTER = ''
1109	020420	017746	173534		MOV	@SWR,-(SP)	::SAVE @SWR FOR TYPEOUT
1110	020424	104402			TYPOC		::GO TYPE--OCTAL ASCII(ALL DIGITS)
1111	020426	104401	014333		TYPE	.\$CRLF	::TYPE CRLF
1112	020432	104401	014333		TYPE	.\$CRLF	
1113	020436	032777	000040	173514	BIT	#UBESW,@SWR	::UBE SWITCH SET?
1114	020444	001411			BEQ	43\$::NOT USED
1115	020446	032777	000200	173504	BIT	#SW07, @SWR	::WILL SECTION 1 BE SKIPPED?
1116	020454	001005			BNE	43\$::BRANCH IF YES
1117	020456	104401	040723		TYPE	.TM4	::'[UNIBUS EXERCISER WILL BE USED]'
1118	020462	105237	016740		INCB	UBEF	::SET UBE FLAG

1119	020466	000404			BR	65\$				
1120	020470	104401	040765		43\$:	TYPE	,TM5	::	'[UNIBUS EXERCISER WILL NOT BE USED]'	
1121	020474	105037	016740			CLRB	UBEF	::	CLEAR THE UBE FLAG	
1122	020500	105737	016737		65\$:	TSTB	MPF	::	MULTIPROCESSOR MODE IN EFFECT?	
1123	020504	001002				BNE	55\$::	BRANCH IF YES	
1124	020506	000137	021076			JMP	STO	::	START SETTING UP VECTORS	
1125					.SBTTL	BOOT AND INITIALIZE THE	SLAVE CPUS			
1126	020512	017702	173442		55\$:	MOV	@SWR,R2	::	COPY SWITCH REGISTER INTO R2	
1127	020516	012737	000001	016774		MOV	#,CPUACT	::	RESET # OF ACTIVE CPUS	
1128	020524	052777	100000	176276		BIS	#BIT15,@ACR	::	INITIALIZE THE IIST.	
1129	020532	012705	002000			MOV	#2000,R5	::	LET IT SETTLE AFTER INIT	
1130	020536	077501				SOB	R5,,			
1131	020540	012777	000001	176262	81\$:	MOV	#1,@ACR	::	ACCESS PGCS REGISTER	
1132	020546	032777	004000	176256		BIT	#BIT11,@ADR	::	IS IT READY	
1133	020554	001771				BEQ	81\$::	NOT YET.	
1134	020556	017705	176246			MOV	@ACR,R5	::	COPY ACR TO R5	
1135	020562	072527	177770			ASH	#-10,R5	::	CREAT PHYSICAL ID	
1136	020566	104401	041665			TYPE	,TM76	::	IDENTIFY THE MASTER	
1137	020572	010546				MOV	R5, -(SP)			
1138	020574	104405				TYPDS				
1139	020576	010537	014226			MOV	R5,\$CPUID	::	SET SELF ID OF MASTER IN TABLE	
1140	020602	005000				CLR	R0	::	REGO. CONTAINS WORD DISPLACE-	
1141								::	MENT INTO CPUID TABLE ***	
1142	020604	005001				CLR	R1	::	R1 CONTAINS THE BYTE DISPLACEMENT,...	
1143								::	THE TRUE LOGICAL ID.	
1144	020606	012777	000007	176214		MOV	#DCF,@ACR	::	ACCESS DCF REGISTER OF IIST	
1145	020614	017703	176212			MOV	@ADR,R3 ::	COPY DCF INTO R3		
1146	020620	072327	177770			ASH	#-10,R3	::	BRING BRK MASK INTO POSITION	
1147	020624	012777	021140	176202		MOV	#SLVENT,@ISTVEC	::	SET ENTRY POINT FOR SLAVES	
1148										
1149	020632	005004				CLR	R4	::	R4 CONTAINS THE DECIMAL VALUE	
1150								::	OF THE SELF ID OF THE CPU UNDER	
1151								::	INTEROGATION.	
1152	020634	032702	000001		1\$:	BIT	#BIT0,R2	::	DO WE WANT THIS CPU?	
1153	020640	001471				BEQ	2\$::	NO,CONTINUE	
1154	020642	032703	000001			BIT	#BIT0,R3	::	IS IT ALIVE?	
1155	020646	001413				BEQ	82\$::	YES	
1156	020650	010437	014310			MOV	R4,\$TMP0	::	SAVE CONTENTS OF R4	
1157	020654	104401	014333			TYPE	,\$CRLF	::	NO,CRLF	
1158	020660	104401	041176			TYPE	,TM11	::	'CPU # ''	
1159	020664	013746	014310			MOV	\$TMP0,-(SP)	::	SAVE \$TMP0 FOR TYPEOUT	
1160	020670	104405				TYPDS		::	GO TYPE--DECIMAL ASCII WITH SIGN	
1161	020672	104401	041207			TYPE	,TM12	::	'SPECIFIED BUT NOT ACTIVE'	
1162	020676	020437	014226		82\$:	CMP	R4,\$CPUID	::	IS THIS THE MASTER?	
1163	020702	001450				BEQ	2\$::	YES, IGNORE	
1164	020704	012737	020000	000000		MO	#20000,@#0	::	M9312MP MOVES @#0 TO SP ON BOOT	
1165	020712	012777	000000	176110		MOV	#PGTE,@ACR	::	ACCESS PGTE REG.	
1166	020720	013777	017040	176104		MOV	BMSK,@ADR	::	SET TO BOOT AND THEN...	
1167	020726	052777	000001	176076		BIS	#BIT0,@ADR	::	BOOT THE CPU	
1168	020734	012701	000200			MOV	#200, R1	::	SET UP A LONG DELAY (5 SEC)	
1169	020740	077001			70\$:	SOB	R0,			
1170	020742	077102				SOB	R1, 70\$			
1171	020744	012777	000001	176056	83\$:	MOV	#PGCS,@ACR	:	CHECK FOR IIST READY	
1172	020752	032777	004000	176052		BIT	#BIT11,@ADR			
1173	020760	001771				BEQ	83\$::	BRANCH IF NOT	
1174	020762	005077	176042		84\$:	CLR	@ACR	::	RESET ACR (POINT TO PGTE)	

```

1175 020766 013777 017000 176036      MOV    INTMSK,@ADR      ;;SET UP TO INTERRUPT
1176 020774 052777 000001 176030      BIS    #BIT0,@ADR      ;;GO!, INTERRUPT SLAVE
1177 021002 005237 016774          INC    CPUACT          ;;COUNT ANOTHER ACTIVE CPU
1178 021006 012777 000001 176014 85$:    MOV    #PGCS,@ACR      ;;CHECK FOR IIST READY
1179 021014 032777 004000 176010      BIT    #BIT11,@ADR
1180 021022 001771          BEQ    85$
1181 021024 020427 000003          2$:    CMP    R4,#3          ;;ALL CPUS STARTED?
1182 021030 002010          BGE    3$            ;;YES
1183 021032 005204          INC    R4            ;;NEXT CPU TO ATTEMPT TO BOOT
1184 021034 006202          ASR    R2            ;;NEXT SWITCH REG. BIT
1185 021036 006203          ASR    R3            ;;NEXT BRK BIT TO SET
1186 021040 006337 017040          ASL    BMSK          ;;NEXT BOOT MASK
1187 021044 006337 017000          ASL    INTMSK        ;;NEXT INTERRUPT MASK
1188 021050 000671          BR     1$            ;;GO TRY ANOTHER
1189 021052 052737 001000 177746 3$:    BIS    #BIT9, CONTRL  ;;SET CACHE BYPASS
1190 021060 005037 000000          CLR    @#0           ;;RESTORE LOC. 0
1191 021064 013737 016774 016770      MOV    CPUACT,CPULS1
1192 021072 005337 016770          DEC    CPULS1        ;CONTAINS # OF ACTIVE SLAVE CPU'S
1193
1194
1195
1196
1197
1198          .SBTTL INITIALIZE THE COMMON TAGS
1199
1200
1201
1202          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1203 021076 052737 000014 177746  STO:   BIS    #14, CONTRL  ;;DISABLE CACHE
1204 021104 012706 014000          MOV    #CMTAG,R6     ;;FIRST LOCATION TO BE CLEARED
1205 021110 005026          CLR    (R6)+         ;;CLEAR MEMORY LOCATION
1206 021112 022706 014160          CMP    #SWR,R6      ;;DONE?
1207 021116 001374          BNE    -6            ;;LOOP BACK IF NO
1208 021120 013706 014200          MOV    $$STP,SP     ;;SETUP THE STACK POINTER
1209 021124 013737 035446 035434      MOV    $ENDCT,$EOPCT ;;SETUP END-OF-PROGRAM COUNTER
1210 021132 005037 014344          CLR    $PASS        ;;CLEAR PASS COUNT
1211 021136 000432          BR    MSTENT
1212 021140
1213 021140 052737 001000 177746  SLVENT: BIS    #BIT9,@CONTRL  ;;TURN OFF CACHE
1214 021146 005237 014644          INC    SLVID         ;;CREATE CPUID
1215 021152 013701 014644          MOV    SLVID,R1     ;;AND MOV TO R1
1216 021156 010100          MOV    R1,R0
1217 021160 006300          ASL    R0            ;;CREATE WORD INDEX INTO CPU TABLE
1218 021162 017705 175642          MOV    @ACR, R5     ;;COPY ACR
1219 021166 072527 177770          ASH    #-10, R5     ;;GET THE ID
1220 021172 010560 014226          MOV    R5,$CPUID(R0) ;;SET SELF-ID INTO TABLE
1221 021176 052777 100000 175624      BIS    #BIT15,@ACR  ;;RESET THE IIST
1222 021204 012777 000001 175616      MOV    #PGCS,@ACR   ;;ENABLE INTERRUPTS
1223 021212 052777 000004 175612      BIS    #BIT2,@ADR
1224 021220 005037 177776          CLR    @PSW         ;;LOWER PROCESSOR PRIORITY
1225 021224
1226 021224 005060 014322          MSTENT: CLR    $ESCAPE(R0)  ;;CLEAR THE ESCAPE(R0) ON ERROR ADDRESS
1227 021230 016060 021242 014032      MOV    10$(R0),$LPERR(R0) ;;SETUP FOR THE ERROR LOOP ADDRESS
1228 021236 000170 021252          JMP    @FORKTB(R0)  ;;DISPATCH THE FOLLOWERS
1229 021242 021450 10$:    TST1
1230 021244 021450          TST1

```

```

1231 021246 021450          TST1
1232 021250 021450          TST1
1233 021252 021342    FORKTB: MS0
1234 021254 021262          SL1
1235 021256 021302          SL2
1236 021260 021322          SL3
1237 021262 016006 014200  SL1:  MOV    $$$STP(R0),SP    ;;INITIALIZE SLAVE STACK (CPU1)
1238 021266 000001          WAIT    ;;WAIT FOR MASTER TO START VIA INTERRUPT
1239 021270 052777 100000 175532  BIS    #BIT15,@ACR    ;;RESET THE IIST
1240 021276 000137 021450          JMP    TST1
1241 021302 016006 014200  SL2:  MOV    $$$STP(R0),SP    ;;INITIALIZE SLAVE STACK (CPU2)
1242 021306 000001          WAIT    ;;WAIT FOR MASTER TO INTERRUPT
1243 021310 052777 100000 175512  BIS    #BIT15,@ACR    ;;RESET THE IIST
1244 021316 000137 021450          JMP    TST1
1245 021322 016006 014200  SL3:  MOV    $$$STP(R0),SP    ;;INITIALIZE SLAVE STACK (CPU3)
1246 021326 000001          WAIT    ;;WAIT FOR MASTER TO INTERRUPT
1247 021330 052777 100000 175472  BIS    #BIT15,@ACR    ;;RESET THE IIST
1248 021336 000137 021450          JMP    TST1
1249 021342 105737 016737  MS0:  TSTB   MPF            ;;MP MODE?
1250 021346 001440          BEQ    BEGIN          ;;NO,DON'T TRY TO INTERRUPT CPUS
1251 021350 013702 017034          MOV    ISTVEC, R2     ;;SET UP RETURN FROM INTERRUPT
1252 021354 062702 000002          ADD    #2, R2
1253 021360 010277 175450          MOV    R2, @ISTVEC
1254 021364 012712 000002          MOV    #2, (R2)
1255 021370 012777 000000 175432  MOV    #PGTE,@ACR     ;ACCESS PGTE REGISTER
1256 021376 017702 172556          MOV    @SWR,R2        ;GET COPY OF SWR
1257 021402 042702 177760          BIC    #177760,R2     ;KEEP ONLY THE CPU MASK
1258 021406 010277 175420          MOV    R2,@ADR        ;SET THE INTERRUPT BITS
1259 021412 032737 000001 016774  BIT    #BIT0,CPUACT   ;EVEN OR ODD?
1260 021420 001404          BEQ    7$             ;BRANCH IF EVEN
1261 021422 012777 000005 175402  MOV    #5,@ADR        ;GO WITH NO PARITY,ENABLE INTERRUPTS
1262 021430 000403          BR     4$
1263 021432 012777 000007 175372  7$:  MOV    #7,@ADR        ;GO WITH PARITY,ENABLE INTS.
1264 021440 000001          4$:  WAIT    ;;WAIT FOR IIST TO INTERRUPT
1265 021442 052777 100000 175360  BIS    #BIT15,@ACR    ;;RESET THE IIST
1266 021450          BEGIN:
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
  
```

1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342

021450
021450 012777 000001 172512
021456 012737 033774 014662
021464 012737 033340 000114
021472 012737 000001 033342
021500 012737 040606 000024
021506 005037 000026
021512 032770 000200 014160
021520 001402
021522 000137 024220
021526 005700
021530 001011
021532 104401 041605
021536 105737 016740
021542 001004
021544 104401 041241
021550 104401 014333
021554
021554 005037 177776
021560 012703 021616
021564 105737 016740
021570 001407
021572 106237 017022
021576 103375
021600 000241
021602 052737 000020 170016
021610
021610 005037 177776
021614 000001
021616 010602
021620 016004 014200
021624 162704 000004
021630 020402
021632 001401
021634 000000
021636 016006 014200
021642 012402
021644 105737 016740
021650 001004
021652 022702 021616

SECTION ONE

*TEST 1 SIMPLE DOWN/UP TEST (KERNAL)

```
TST1:
MOV #1,@DISPLAY ;SET TEST NUMBER
MOV #POWDWN, PWRTAB ;SET UP POWER DOWN VECTOR
MOV #PARERR, @CACHVEC ;SET PARITY ERROR VECTOR
MOV #1, @PARFLG ;SET MULTI PARITY ERROR INDICATOR
MOV #PWRDIS,@PWRVEC ;SET LOC 24
CLR @PWRVEC+2 ;SET LOC 26
BIT #SW07, @SWR(R0) ;SKIP SECTION 1?
BEQ 1$ ;BRANCH IF NO
JMP SEC2 ;ELSE GO TO SEC2
1$: TST R0 ;IS THIS THE MASTER?
BNE 4$ ;BRANCH IF NO
TYPE ,TM14 ;'ENTERING SECTION 1''
TSTB UBEF ;USING THE UBE?
BNE 4$ ;BRANCH IF YES
TYPE ,TM13 ;PRINT INSTRUCTIONS
TYPE ,$CRLF

4$: CLR @MPS ;SET KERNAL MODE
MOV #2$,R3 ;SET POWER UP RETURN
TSTB UBEF ;USE UNIBUS EXERCISER?
BEQ 64$ ;BRANCH IF NO
ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING
BCC -4
CLC
BIS #BIT4,@UBCR2 ;;SET TO POWER FAIL

64$: CLR @MPS ;SET KERNAL MODE
3$: WAIT ;WAIT FOR THE POWER FAIL
2$: MOV SP,R2 ;GET SP
MOV $$STP(R0),R4 ;R4 CONTAINS THE STACK INIT. VALUE
SUB #4,R4 ;STACK-4
CMP R4,R2 ;CHECK STACK
BEQ .+4 ;SKIP IF OK
HALT ;SP NOT 'STACK-4''
MOV $$STP(R0),SP ;RESET SP
MOV (R4)+,R2 ;GET RETURN ADDRESS
TSTB UBEF ;IS THE UBE BEING USED?
BNE 72$ ;YES
70$: CMP #2$,R2 ;CHECK ADDRESS
```


1343 021656 001401
 1344 021660 000000
 1345 021662 011402
 1346 021664 022702 000000
 1347 021670 001401
 1348 021672 000000
 1349 021674 032770 040000 014160
 1350 021702 001262

BEQ .+4 ;SKIP IF OK
 HALT ;ADDRESS ON STACK IS WRONG
 72\$: MOV (R4),R2 ;GET OLD PS
 CMP #0,R2 ;CHECK OLD PS
 BEQ .+4 ;SKIP IF OK
 HALT ;OLD PS IS WRONG
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST1 ;LOOP TO TST1

 :*TEST 2 PROGRAM VOLATILITY TEST

1356 021704
 1357 021704 012777 000002 172256
 1358 021712 005037 177776
 1359 021716 012702 010000
 1360 021722 012703 020000
 1361 021726 005060 014650
 1362 021732 062360 014650
 1363 021736 005560 014650
 1364 021742 077205
 1365 021744 012703 022000
 1366 021750 105737 016740
 1367 021754 001407
 1368 021756 106237 017022
 1369 021762 103375
 1370 021764 000241
 1371 021766 052737 000020 170016
 1372 021774
 1373 021774 000001
 1374 021776 000000
 1375 022000 012702 010000
 1376 022004 012703 020000
 1377 022010 005004
 1378
 1379 022012 062304
 1380 022014 005504
 1381 022016 077203
 1382 022020 020460 014650
 1383
 1384 022024 001401
 1385 022026 000000
 1386 022030 016006 014200
 1387 022034 032770 040000 014160
 1388 022042 001320

TST2:
 MOV #2,@DISPLAY ;SET TEST NUMBER
 CLR @#PS ;SET KERNAL MODE
 MOV #10000,R2 ;INIT. COUNTER
 MOV #20000,R3 ;INIT POINTER
 CLR CKSUM(R0) ;RESET THE CHECKSUM LOCATION
 1\$: ADD (R3)+,CKSUM(R0) ;DO CHECKSUM ON 2ND 4K(W) BANK
 ADC CKSUM(R0)
 SOB R2,1\$
 MOV #2\$,R3 ;POWER UP RETURN
 TSTB UBEF ;USE UNIBUS EXERCISER?
 BEQ 64\$;BRANCH IF NO
 ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING
 BCC -4
 CLC
 BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL
 64\$: WAIT ;WAIT FOR THE POWER TO FAIL
 HALT ;BAD
 2\$: MOV #10000,R2
 MOV #20000,R3
 CLR R4
 ;VERIFY THAT EVERYTHING IS OK
 3\$: ADD (R3)+,R4
 ADC R4
 SOB R2,3\$
 CMP R4,CKSUM(R0) ;COMPARE NEW CHECKSUM WITH OLD
 BEQ 5\$;BRANCH IF OK
 HALT ;ERROR
 5\$: MOV \$\$STP(R0),SP ;RESET THE STACK
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST2 ;LOOP TO TST2

 :*TEST 3 SIMPLE DOWN/UP TEST (SUPERVISOR)

1392 022044
 1393 022044 012777 000003 172116
 1394 022052 012737 040000 177776
 1395 022060 012703 022120
 1396 022064 105737 016740
 1397 022070 001407
 1398 022072 106237 017022

TST3:
 MOV #3,@DISPLAY ;SET TEST NUMBER
 MOV #40000,@#PS ;SET SUPERVISOR MODE
 MOV #2\$,R3 ;SET POWER UP RETURN
 TSTB UBEF ;USE UNIBUS EXERCISER?
 BEQ 64\$;BRANCH IF NO
 ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING

```

1399 022076 103375          BCC      .-4
1400 022100 000241          CLC
1401 022102 052737 000020 170016 64$:    BIS      #BIT4,@#UBCR2  ;;SET TO POWER FAIL
1402 022110          3$:
1403 022110          MOV      #40000,@#PS    ;SET SUPERVISOR MODE
1404 022110 012737 040000 177776 3$:    WAIT          ;WAIT FOR THE POWER FAIL
1405 022116 000001          MOV      $$STP(R0),SP  ;RESET
1406 022120 016006 014200 2$:    MOV      $$STP(R0),R4
1407 022124 016004 014200          MOV      #4,R4
1408 022130 162704 000004          SUB      (R4)+,R2      ;GET RETURN ADDRESS
1409 022134 012402          MOV      UBEF
1410 022136 105737 016740          TSTB
1411 022142 001004          BNE     72$
1412 022144 022702 022120 70$:    CMP      #2$,R2      ;CHECK ADDRESS
1413 022150 001401          BEQ     .+4          ;SKIP IF OK
1414 022152 000000          HALT          ;ADDRESS ON STACK IS WRONG
1415 022154 011402          MOV     72$:    (R4),R2      ;GET OLD PS
1416 022156 022702 040000          CMP     #40000,R2    ;CHECK OLD PS
1417 022162 001401          BEQ     .+4          ;SKIP IF OK
1418 022164 000000          HALT          ;OLD PS IS WRONG
1419 022166          1$:
1420 022166 032770 040000 014160 BIT     #SW14,@SWR(R0) ;LOOP ON TEST?
1421 022174 001323          BNE     TST3        ;LOOP TO TST3
1422
1423
1424
1425
1426
1427 022176          :*****
1428 022176 012777 000004 171764 :*TEST 4      SIMPLE DOWN/UP TEST (USER)
1429 022204 012737 140000 177776 :*****
1430 022212 012703 022252          TST4:
1431 022216 105737 016740          MOV     #4,@DISPLAY  ;SET TEST NUMBER
1432 022222 001407          MOV     #140000,@#PS ;SET USER MODE
1433 022224 106237 017022          MOV     #2$,R3      ;SET POWER UP RETURN
1434 022230 103375          TSTB   UBEF         ;USE UNIBUS EXERCISER?
1435 022232 000241          BEQ     64$         ;BRANCH IF NO
1436 022234 052737 000020 170016 ASRB   UBELCK       ;LOCK OUT OTHER CPUS FROM PROCEEDING
1437 022242          BCC     .-4
1438 022242          CLC
1439 022242 012737 140000 177776 64$:    BIS     #BIT4,@#UBCR2 ;;SET TO POWER FAIL
1440 022250 000001          3$:
1441 022252 016006 014200 2$:    MOV     #140000,@#PS ;SET USER MODE
1442 022256 016004 014200          WAIT          ;WAIT FOR THE POWER FAIL
1443 022262 162704 000004          MOV     $$STP(R0),SP ;RESET SP
1444 022266 012402          MOV     $$STP(R0),R4 ;GET STACK INIT. VALUE
1445 022270 105737 016740          SUB     #4,R4       ;MINUS 4
1446 022274 001004          MOV     (R4)+,R2    ;GET STACK-4,AUTOINC. STACK
1447 022276 022702 022252 70$:    TSTB   UBEF
1448 022302 001401          BNE     72$
1449 022304 000000          CMP     #2$,R2     ;CHECK ADDRESS
1450 022306 011402          BEQ     .+4        ;SKIP IF OK
1451 022310 022702 140000          HALT          ;ADDRESS ON STACK IS WRONG
1452 022314 001401          MOV     72$:    (R4),R2 ;GET OLD PS
1453 022316 000000          CMP     #140000,R2 ;CHECK OLD PS
1454 022320 032770 040000 014160 BEQ     .+4        ;SKIP IF OK
          HALT          ;OLD PS IS WRONG
          BIT     #SW14,@SWR(R0) ;LOOP ON TEST?
    
```

1455 022326 001323 BNE TST4 ;LOOP TO TST4

1456
1457
1458
1459

: *TEST 5 POWER FAIL WITH ODD ADDRESS

1460
1461

1462 022330

TST5:

1463 022330 012777 000005 171632

MOV #5,@DISPLAY ;SET TEST NUMBER

1464 022336 005037 177776

CLR @MPS ;SET KERNAL MODE

1465 022342 012760 022400 014702

MOV #3\$,ERRTAB(R0) ;SET TRAP VECTOR

1466 022350 012703 022426

MOV #1\$,R3 ;SET RETURN ADDRESS FOR POWER FAIL

1467 022354 105737 016740

TSTB UBEF ;:USE UNIBUS EXERCISER?

1468 022360 001407

BEQ 64\$;:BRANCH IF NO

1469 022362 106237 017022

ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING

1470 022366 103375

BCC -4

1471 022370 000241

CLC

1472 022372 052737 000020 170016

BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL

1473 022400

64\$:

1474 022400 016006 014200

MOV \$\$STP(R0),SP ;RESET STACK

1475 022404 005737 000003

3\$:

TST @#3 ;CAUSE ODD ADDRESS TRAP

1476 022410 105737 016740

TSTB UBEF ;:USE UNIBUS EXERCISER?

1477 022414 001403

BEQ 65\$

1478 022416 042737 000020 170016

BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE

1479 022424

65\$:

1480 022424 000000

HALT ;ODD ADDRESS TRAP FAILED

1481 022426 012760 040630 014702

1\$:

MOV #CPUER,ERRTAB(R0) ;RESET 4

1482 022434 032770 040000 014160

BIT #SW14,@SWR(R0) ;LOOP ON TEST?

1483 022442 001332

BNE TST5 ;LOOP TO TST5

1484

1485

1486

1487

1488

1489

1490

1491 022444

: *TEST 6 POWER FAIL IN THE RED ZONE

1492 022444 012777 000006 171516

TST6:

1493 022452 005037 177776

MOV #6,@DISPLAY ;SET TEST NUMBER

1494 022456 012760 022526 014702

CLR @MPS ;SET KERNAL MODE

1495 022464 012703 022560

MOV #2\$,ERRTAB(R0) ;SET TRAP REGISTER

1496 022470 012706 000002

MOV #1\$,R3 ;SET POWER UP RETURN

1497 022474 105737 016740

MOV #2,SP ;SET STACK TO RED ZONE

1498 022500 001407

TSTB UBEF ;:USE UNIBUS EXERCISER?

1499 022502 106237 017022

BEQ 64\$;:BRANCH IF NO

1500 022506 103375

ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING

1501 022510 000241

BCC -4

1502 022512 052737 000020 170016

BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL

1503 022520

64\$:

1504 022520 005037 177776

CLR @MPS ;SET KERNAL MODE

1505 022524 000001

WAIT ;WAIT FOR POWER FAIL TRAP

1506 022526 012737 022534 014662

2\$:

MOV #7\$,PWRTAB ;SET UVEC TO HALT

1507 022534

7\$:

1508 022534 105737 016740

TSTB UBEF ;:USE UNIBUS EXERCISER?

1509 022540 001403

BEQ 65\$

1510 022542 042737 000020 170016

BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE

1511 022550
 1512 022550 000000
 1513 022552 012737 033774 014662
 1514 022560 016006 014200
 1515 022564 012760 040630 014702
 1516 022572 013702 000002
 1517 022576 023727 000000
 1518 022604 001401
 1519 022606 000000
 1520 022610 013702 000000
 1521 022614 022737 040606 000000
 1522 022622 001401
 1523 022624 000000
 1524 022626 032770 040000 014160
 1525 022634 001303

65\$:
 HALT ;ILLEGAL TRAP TO 4
 MOV #POWDN,PWRTAB ;RESET DVEC
 1\$: MOV \$\$STP(R0),SP ;RESET STACK
 MOV #CPUER,ERRTAB(R0) ;RESET 4
 MOV @#2,R2 ;GET FOR TYPING
 CMP @#2,#0 ;IS 2 OK?
 BEQ .+4 ;SKIP IF OK
 HALT ;NO!
 MOV @#0,R2 ;GET FOR TYPING
 CMP #PWRDIS,@#0 ;IS 0 OK?
 BEQ .+4 ;SKIP IF OK
 HALT ;0 IS WRONG.
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST6 ;LOOP TO TST6

 : *TEST 7 POWER FAIL WITH TIME OUT (KERNAL)

1531 022636
 1532 022636 012777 000007 171324
 1533 022644 005037 177776
 1534 022650 012760 022706 014702
 1535 022656 012703 022740
 1536 022662 105737 016740
 1537 022666 001407
 1538 022670 106237 017022
 1539 022674 103375
 1540 022676 000241
 1541 022700 052737 000020 170016
 1542 022706
 1543 022706 016006 014200
 1544 022712 005037 177776
 1545 022716 010037 173000
 1546 022722 105737 016740
 1547 022726 001403
 1548 022730 042737 000020 170016
 1549 022736
 1550 022736 000000
 1551 022740 016006 014200
 1552 022744 012760 040630 014702
 1553 022752 032770 040000 014160
 1554 022760 001326

TST7:
 MOV #7,@DISPLAY ;SET TEST NUMBER
 CLR @#PS ;SET KERNAL MODE
 MOV #3\$,ERRTAB(R0) ;SET TRAP VECTOR
 MOV #1\$,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL
 TSTB UBEF ;:USE UNIBUS EXERCISER?
 BEQ 64\$;:BRANCH IF NO
 ASRB UBELCK ;:LOCK OUT OTHER CPUS FROM PROCEEDING
 BCC .-4
 CLC
 BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL
 64\$:
 3\$: MOV \$\$STP(R0),SP ;SET STACK
 CLR @#PS ;SET KERNAL MODE
 MOV R0,@#173000 ;CAUSE A TIMEOUT
 TSTB UBEF ;:USE UNIBUS EXERCISER?
 BEQ 65\$
 BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE
 65\$:
 HALT ;TIMEOUT FAILED
 1\$: MOV \$\$STP(R0),SP ;SET STACK
 MOV #CPUER,ERRTAB(R0) ;RESET 4
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST7 ;LOOP TO TST7

 : *TEST 10 POWER FAIL IN THE YELLOW ZONE (KERNAL)

1560 022762
 1561 022762 012777 000010 171200
 1562 022770 005037 177776
 1563 022774 005037 014602
 1564 023000 012760 023070 014702
 1565 023006 012706 000400
 1566 023012 012703 023050

TST10:
 MOV #10,@DISPLAY ;SET TEST NUMBER
 CLR @#PS ;SET KERNAL MODE
 CLR FLAG ;CLEAR THE FLAG
 MOV #2\$,ERRTAB(R0) ;SET SICK TPAP ADDRESS
 MOV #400,SP ;SET STACK TO YELLOW ZONE
 MOV #1\$,R3 ;SET RETURN ADDRESS FOR POWER FAIL

```

1567 023016 105737 016740      TSTB  UBEF      ;;USE UNIBUS EXERCISER?
1568 023022 001407      BEQ   64$      ;;BRANCH IF NO
1569 023024 106237 017022      ASRB  UBELCK   ;LOCK OUT OTHER CPUS FROM PROCEEDING
1570 023030 103375      BCC   -4
1571 023032 000241      CLC
1572 023034 052737 000020 170016      BIS   #BIT4,@#UBCR2 ;;SET ) POWER FAIL
1573 023042      64$:
1574 023042 005037 177776      CLR   @#PS     ;SET KERNAL MODE
1575 023046 000001      WAIT                    ;WAIT FOR POWER FAIL
1576 023050      1$:
1577 023050 105737 016740      TSTB  UBEF      ;;USE UNIBUS EXERCISER?
1578 023054 001403      BEQ   65$
1579 023056 042737 000020 170016      BIC   #BIT4,@#UBCR2 ;;CLEAR POWER FAIL ENABLE
1580 023064      65$:
1581 023064 000000      HALT                    ;POWER FAIL RETURNED TOO SOON
1582 023066 000430      BR    4$             ;SKIP SP CHECK
1583 023070 012760 040630 014702      2$:  MOV   #CPUER,ERRTAB(R0) ;RESET 4
1584 023076 005737 014602      TST   FLAG         ;IS THE FIRST INSTRUCTION FLAG SET?
1585 023102 001016      BNE   5$             ;YES
1586 023104 012737 023112 014662      MOV   #7$,PWRTAB   ;SET UVEC TO HALT
1587 023112      7$:
1588 023112 105737 016740      TSTB  UBEF      ;;USE UNIBUS EXERCISER?
1589 023116 001403      BEQ   66$
1590 023120 042737 000020 170016      BIC   #BIT4,@#UBCR2 ;;CLEAR POWER FAIL ENABLE
1591 023126      66$:
1592 023126 000000      HALT                    ;NOT ENOUGH OR TOO MANY INSTR. EXEC.
1593 023130 012737 033774 014662      MOV   #POWDWN,PWRTAB ;SET DVEC
1594 023136 000404      BR    4$             ;GET OUT
1595 023140 012703 023150      5$:  MOV   #4$,R3        ;SET RETURN
1596 023144 000002      RTI                    ;GO TO THE POWER FAIL ROUTINE
1597 023146 000000      HALT                    ;SHOULD NOT RETURN HERE
1598 023150      4$:
1599 023150 032770 040000 014160      BIT   #SW14,@SWR(R0) ;LOOP ON TEST?
1600 023156 001301      BNE  TST10          ;LOOP TO TST10

```

```

1601
1602
1603      ;;*****
1604      ;*TEST 11      POWER FAIL WITH RESETS
1605      ;;*****
1606      TST11:
1607 023160 012777 000011 171002      MOV   #11,@DISPLAY   ;SET TEST NUMBER
1608 023166 005037 177776      CLR   @#PS           ;SET KERNAL MODE
1609 023172 012703 023236      MOV   #1$,R3         ;SET RETURN ADDRESS
1610 023176 016006 014200      MOV   $$STP(R0),SP   ;RESET STACK
1611 023202 105737 016740      TSTB  UBEF      ;;USE UNIBUS EXERCISER?
1612 023206 001407      BEQ   64$      ;;BRANCH IF NO
1613 023210 106237 017022      ASRB  UBELCK   ;LOCK OUT OTHER CPUS FROM PROCEEDING
1614 023214 103375      BCC   -4
1615 023216 000241      CLC
1616 023220 052737 000020 170016      BIS   #BIT4,@#UBCR2 ;;SET TO POWER FAIL
1617 023226      64$:
1618 023226 000005      3$:  RESET                    ;RESETS
1619 023230 000005      RESET                    ;TO WAIT
1620 023232 000005      RESET                    ;IN
1621 023234 000774      BR    3$             ;LOOP
1622 023236 016006 014200      1$:  MOV   $$STP(R0),SP   ;RESET STACK

```

1623 023242 032770 040000 014160
1624 023250 001343
1625
1626
1627
1628
1629
1630 023252
1631 023252 012777 000012 170710
1632 023260 012737 040000 177776
1633 023266 012760 023324 014702
1634 023274 012703 023364
1635 023300 105737 016740
1636 023304 001407
1637 023306 106237 017022
1638 023312 103375
1639 023314 000241
1640 023316 052737 000020 170016
1641 023324
1642 023324 016006 014200
1643 023330 012737 040000 177776
1644 023336 005737 000003
1645 023342 005037 177776
1646 023346 105737 016740
1647 023352 001403
1648 023354 042737 000020 170016
1649 023362
1650 023362 000000
1651 023364 016006 014200
1652 023370 012760 040630 014702
1653 023376 032770 040000 014160
1654 023404 001322
1655
1656
1657

```
BIT #SW14,@SWR(R0) ;LOOP ON TEST?  
BNE TST11 ;LOOP TO TST11  
  
:*****  
:*TEST 12 POWER FAIL WITH ODD ADDRESS (SUPERVISOR)  
:*****  
TST12:  
MOV #12,@DISPLAY ;SET TEST NUMBER  
MOV #40000,@#PS ;SET SUPERVISOR MODE  
MOV #3$,ERRTAB(R0) ;SET TRAP VECTOR  
MOV #1$,R3 ;SET RETURN ADDRESS FOR POWER FAIL  
TSTB UBEF ;:USE UNIBUS EXERCISER?  
BEQ 64$ ;:BRANCH IF NO  
ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING  
BCC -4  
CLC  
BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL  
64$:  
3$: MOV $$STP(R0),SP ;RESET STACK  
MOV #40000,@#PS ;SET SUPERVISOR MODE  
TST @#3 ;CAUSE ODD ADDRESS TRAP  
CLR @#PS ;SET KERNAL MODE  
TSTB UBEF ;:USE UNIBUS EXERCISER?  
BEQ 65$  
BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE  
65$:  
1$: HALT ;ODD ADDRESS TRAP FAILED  
MOV $$STP(R0),SP ;RESET STACK POINTER  
MOV #CPUER,ERRTAB(R0) ;RESET 4  
BIT #SW14,@SWR(R0) ;LOOP ON TEST?  
BNE TST12 ;LOOP TO TST12
```

1658
1659
1660 023406
1661 023406 012777 000013 170554
1662 023414 012737 040000 177776
1663 023422 012760 023460 014702
1664 023430 012703 023520
1665 023434 105737 016740
1666 023440 001407
1667 023442 106237 017022
1668 023446 103375
1669 023450 000241
1670 023452 052737 000020 170016
1671 023460
1672 023460 016006 014200
1673 023464 012737 040000 177776
1674 023472 010037 173000
1675 023476 005037 177776
1676 023502 105737 016740
1677 023506 001403
1678 023510 042737 000020 170016

```
:*****  
:*TEST 13 POWER FAIL WITH TIME OUT (SUPERVISOR)  
:*****  
TST13:  
MOV #13,@DISPLAY ;SET TEST NUMBER  
MOV #40000,@#PS ;SET SUPERVISOR MODE  
MOV #3$,ERRTAB(R0) ;SET TRAP VECTOR  
MOV #1$,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL  
TSTB UBEF ;:USE UNIBUS EXERCISER?  
BEQ 64$ ;:BRANCH IF NO  
ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING  
BCC -4  
CLC  
BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL  
64$:  
3$: MOV $$STP(R0),SP ;RESET STACK  
MOV #40000,@#PS ;SET SUPERVISOR MODE  
MOV R0,@#173000 ;CAUSE A TIMEOUT  
CLR @#PS ;SET KERNAL MODE  
TSTB UBEF ;:USE UNIBUS EXERCISER?  
BEQ 65$  
BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE
```

1679 023516
 1680 023516 000000
 1681 023520 016006 014200
 1682 023524 012760 040630 014702
 1683 023532 032770 040000 014160
 1684 023540 001322

65\$:
 HALT ;TIMEOUT FAILED
 1\$: MOV \$\$STP(R0),SP ;RESET STACK
 MOV #CPUER,ERRTAB(R0) ;RESET 4
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST13 ;LOOP TO TST13

1685
 1686
 1687
 1688
 1689

 *TEST 14 POWER FAIL WITH ODD ADDRESS (USER)

1690 023542
 1691 023542 012777 000014 170420
 1692 023550 012737 140000 177776
 1693 023556 012760 023614 014702
 1694 023564 012703 023654
 1695 023570 105737 016740
 1696 023574 001407
 1697 023576 106237 017022
 1698 023602 103375
 1699 023604 000241
 1700 023606 052737 000020 170016

TST14:
 MOV #14,@DISPLAY ;SET TEST NUMBER
 MOV #140000,@#PS ;SET USER MODE
 MOV #3\$,ERRTAB(R0) ;SET TRAP VECTOR
 MOV #1\$,R3 ;SET RETURN ADDRESS FOR POWER FAIL
 TSTB UBEF ;:USE UNIBUS EXERCISER?
 BEQ 64\$;:BRANCH IF NO
 ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING
 BCC -4
 CLC
 BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL

1701 023614
 1702 023614 016006 014200
 1703 023620 012737 140000 177776
 1704 023626 005737 000003
 1705 023632 005037 177776
 1706 023636 105737 016740
 1707 023642 001403
 1708 023644 042737 000020 170016
 1709 023652

64\$:
 3\$: MOV \$\$STP(R0),SP ;RESET STACK
 MOV #140000,@#PS ;SET USER MODE
 TST @#3 ;CAUSE ODD ADDRESS TRAP
 CLR @#PS ;SET KERNAL MODE
 TSTB UBEF ;:USE UNIBUS EXERCISER?
 BEQ 65\$
 BIC #BIT4,@#UBCR2 ;:CLEAR POWER FAIL ENABLE

1710 023652 000000
 1711 023654 016006 014200
 1712 023660 012760 040630 014702
 1713 023666 032770 040000 014160
 1714 023674 001322

65\$:
 1\$: HALT ;ODD ADDRESS TRAP FAILED
 MOV \$\$STP(R0),SP ;RESET SP
 MOV #CPUER,ERRTAB(R0) ;RESET 4
 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
 BNE TST14 ;LOOP TO TST14

1715
 1716
 1717
 1718
 1719

 *TEST 15 POWER FAIL WITH TIME OUT (USER)

1720 023676
 1721 023676 012777 000015 170264
 1722 023704 012737 140000 177776
 1723 023712 012760 023750 014702
 1724 023720 012703 024010
 1725 023724 105737 016740
 1726 023730 001407
 1727 023732 106237 017022
 1728 023736 103375
 1729 023740 000241
 1730 023742 052737 000020 170016
 1731 023750
 1732 023750 016006 014200
 1733 023754 012737 140000 177776
 1734 023762 010C37 173000

TST15:
 MOV #15,@DISPLAY ;SET TEST NUMBER
 MOV #140000,@#PS ;SET USER MODE
 MOV #3\$,ERRTAB(R0) ;SET TRAP VECTOR
 MOV #1\$,R3 ;SET UP RETURN ADDRESS FOR POWER FAIL
 TSTB UBEF ;:USE UNIBUS EXERCISER?
 BEQ 64\$;:BRANCH IF NO
 ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING
 BCC -4
 CLC
 BIS #BIT4,@#UBCR2 ;:SET TO POWER FAIL

64\$:
 3\$: MOV \$\$STP(R0),SP ;RESET STACK
 MOV #140000,@#PS ;SET USER MODE
 MOV R0,@#173000 ;CAUSE A TIMEOUT

```

1735 023766 005037 177776 CLR @MPS ;SET KERNAL MODE
1736 023772 105737 016740 TSTB UBEF ;;USE UNIBUS EXERCISER?
1737 023776 001403 BEQ 65$
1738 024000 042737 000020 170016 BIC #BIT4,@#UBCR2 ;;CLEAR POWER FAIL ENABLE
1739 024006 65$:
1740 024006 000000 HALT ;TIMEOUT FAILED
1741 024010 016006 014200 1$: MOV $$STP(R0),SP ;RESET STACK
1742 024014 012760 040630 014702 MOV #CPUER,ERRTAB(R0) ;RESET 4
1743 024022 032770 040000 014160 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
1744 024030 001322 BNE TST15 ;LOOP TO TST15
1745
1746
1747
1748 ;*****
1749 ;*TEST 16 MEMORY MANAGEMENT ABORT TEST
1750 ;*****
1750 024032 TST16:
1751 024032 012777 000016 170130 MOV #16,@DISPLAY ;SET TEST NUMBER
1752 024040 005037 177776 CLR @MPS ;SET KERNAL MODE
1753 024044 012760 004000 014612 MOV #TI, PFFT(R0) ;TIME THIS POWER FAIL
1754 024052 012760 024152 014702 MOV #4$,ERRTAB(R0) ;SET FOR TIMEOUT
1755 024060 004737 033406 JSR PC,MAP ;MAP THE WORLD
1756 024064 012737 024126 000250 MOV #3$,@MMVEC ;SET MEMORY MANAGEMENT VECTOR
1757 024072 012703 024154 MOV #1$,R3 ;LOAD PF RETURN
1758 024076 005237 177572 INC @MMR0 ;TURN MEMORY MANAGEMENT ON
1759 024102 105737 016740 TSTB UBEF ;;USE UNIBUS EXERCISER?
1760 024106 001407 BEQ 64$ ;;BRANCH IF NO
1761 024110 106237 017022 ASRB UBELCK ;LOCK OUT OTHER CPUS FROM PROCEEDING
1762 024114 103375 BCC .-4
1763 024116 000241 CLC
1764 024120 052737 000020 170016 BIS #BIT4,@#UBCR2 ;;SET TO POWER FAIL
1765 024126 64$:
1766 024126 016006 014200 3$: MOV $$STP(R0),SP ;ZAP STACK
1767 024132 005237 140000 INC @#140000 ;ACCESS VIOLATION
1768 024136 105737 016740 TSTB UBEF ;;USE UNIBUS EXERCISER?
1769 024142 001403 BEQ 65$
1770 024144 042737 000020 170016 BIC #BIT4,@#UBCR2 ;;CLEAR POWER FAIL ENABLE
1771 024152 65$:
1772 024152 000000 4$: HALT ;NO VIOLATION OR TRAP TO 4
1773
1774 024154 005037 177572 1$: CLR @MMR0 ;TURN OFF MEMORY MANAGEMENT
1775 024160 016006 014200 2$: MOV $$STP(R0),SP ;MAKE A NEW STACK
1776 024164 012760 040630 014702 MOV #CPUER,ERRTAB(R0) ;RESET 4
1777 024172 032770 040000 014160 BIT #SW14,@SWR(R0) ;LOOP ON TEST?
1778 024200 001314 BNE TST16 ;LOOP TO TST16
1779 024202 005077 167762 CLR @DISPLA ;CLEAR THE DISPLAY REGISTER.
1780 024206 105737 016737 TSTB MPF ;MP MODE?
1781 024212 001002 BNE 5$ ;BRANCH IF YES
1782 024214 000137 035326 JMP $EOP ;UUMP INTO EOP
1783 024220 5$:
1784
1785
1786
1787
1788
1789
1790

```


1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846

```
*****  
*****  
*****  
***** SECTION TWO *****  
*****  
*****  
*****
```

```
SEC2: .SBTTL SECTION 2 INITIALIZATION  
MOV #1$, R3 ;SET UP POWER FAIL RETURN IN CASE  
ASRB SYNC.3 ;CONTROL THE ENTRY  
BCC SEC2  
INC S2LOG1 ;LOG INTO SEC2  
MOV #1, SYNC.3 ;LET THE OTHERS IN  
1$: CMP S2LOG1,CPUACT ;:WAIT FOR EVERYONE TO GET ...  
BNE 1$ ;:HERE  
TST R0 ;:IS THIS THE MASTER?  
BNE 3$ ;:BRANCH IF NO  
TYPE .TM15 ;:'ENTERING SECTION 2'  
3$:  
BIS #BIT15,@ACR ;INITIALIZE IIST  
MOV $$STP(R0), SP ;SET THE STACK  
2$: ASRB S2L1 ;:TRY TO ENTER SECTION 2 INITIALIZATION  
BCC 2$  
MOV #POWER,@#24 ;:SET NEW POWER FAIL HANDLER  
MOV #340,@#26  
MOV @ACR, R5 ;COPY ACR  
ASH #-10, R5 ;GET THE ID  
MOV R5, $REGO(R0);IDENTIFY CPU FOR ERROR TYPE-OUT  
CLR R2 ;RESET FOR COUNT  
65$: CMP $CPUID(R2),R5 ;SID MATCH?  
BEQ 64$  
TST (R2)+ ;INCREMENT R2 BY 2  
CMP R2,#10  
BLT 65$  
64$: MOV R2,R0 ;MOV LOGICAL ID TO 2ND OPERAND  
MOV R0, R1 ;SET UP R1  
ASR R1  
MOV #SPWRDN, PWRTAB(R0) ;SET UP FOR POWER DOWN
```

1847	024370	012760	024664	014702	SIZMBS:	MOV	#NORP,ERRTAB(R0)	::SET UP CPU ERROR VECTOR IN CASE
1848								::THERE'S NO RPO/5/6
1849	024376	052737	000040	176710		BIS	#BIT5,@#RPCS2	::INIT. RP CONTROLLER,IF THERE.
1850	024404	012760	040630	014702		MOV	#CPUER,ERRTAB(R0)	::RESET ERROR VECTOR
1851	024412	005002				CLR	R2	::RESET COUNTER
1852	024414	010237	176710		RPSRC:	MOV	R2,@#RPCS2	::SET DRIVE # IN CS REG.
1853	024420	032737	040000	176712		BIT	#BIT14,@#RPDS	::IS THE DRIVE UP?
1854	024426	001024				BNE	NXTDRV	::BRANCH IF IT ISN'T
1855								
1856	024430	032737	001000	176712		BIT	#BIT9,@#RPDS	::IS THE PGM BIT SET FOR THIS DRIVE
1857	024436	001403				BEQ	1\$	
1858	024440	005260	016742			INC	RPPGM(R0)	::YES, FLAG THE CONDITION
1859	024444	000415				BR	NXTDRV	::AND, SEARCH FOR ANOTHER DRIVE
1860	024446	032737	000400	176712	1\$:	BIT	#BIT8,@#RPDS	::IS THIS PORT IN CONTROL?
1861	024454	001411				BEQ	NXTDRV	::NO, LOOK FOR ANOTHER
1862	024456	005060	014632			CLR	MBDSW(R0)	::CLEAR MASSBUSS DEVICE SELECTION
1863	024462	013703	176710			MOV	@#RPCS2,R3	::COPY CS2
1864	024466	042703	177770			BIC	#177770,R3	::GET RID OF OTHER INFO
1865	024472	110360	014632			MOVB	R3,MBDSW(R0)	::WRITE DRIVE ID INTO SELECTION W
1866	024476	000454				BR	SIZEND	::DEVICE HAS BEEN FOUND.
1867	024500	005202			NXTDRV:	INC	R2	::NEXT DRIVE
1868	024502	020227	000010			CMP	R2,#10	::ALL DRIVES TESTED?
1869	024506	103742				BLO	RPSRC	::NO, TEST SOME MORE.
1870	024510	005760	016742			TST	RPPGM(R0)	::ANY PROGRAMMABLE DRIVES?
1871	024514	001011				BNE	A	::YES.
1872	024516	104401	041750		NORH70:	TYPE	,TM101	::'NO MASSBUS DEVICE AVAILABLE ON CPU #'
1873	024522	016046	014226			MOV	\$CPUID(R0),-(SP)	::SAVE \$CPUID(R0) FOR TYPEOUT
1874	024526	104405				TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
1875	024530	012760	001400	014632		MOV	#1400,MBDSW(R0)	::SET CODE FOR NO DEVICE
1876	024536	000434				BR	SIZEND	::EXIT SECTION 2 INITIALIZATION
1877	024540	005002			A:	CLR	R2	::RESET COUNTER
1878	024542	010237	176710		10\$:	MOV	R2,@#RPCS2	::ACCESS DRIVE.
1879	024546	032737	001000	176712		BIT	#BIT9,@#RPDS	::PGM BIT SET?
1880	024554	001005				BNE	15\$::YES, FOUND ONE
1881	024556	005202				INC	R2	::NO, NEXT DRIVE
1882	024560	020227	000010			CMP	R2,#10	::ALL DRIVE TESTED
1883	024564	103766				BLO	10\$::NO.
1884	024566	000000				HALT		::YES.
1885	024570	110260	014632		15\$:	MOVB	R2,MBDSW(R0)	::SET DRIVE #
1886	024574	052760	001000	014632		BIS	#BIT9,MBDSW(R0)	::SET PGM BIT
1887	024602	104401	043752			TYPE	,SPGM1	::TYPE SHARED DRIVE WARNING MSG
1888	024606	010246				MOV	R2, -(SP)	
1889	024610	104405				TYPDS		
1890	024612	104401	044000			TYPE	,SPGM2	
1891	024616	016046	014226			MOV	\$CPUID(R0), -(SP)	
1892	024622	104405				TYPDS		
1893	024624	104401	044014			TYPE	,SPGM3	
1894	024630	012737	000001	017026	SIZEND:	MOV	#1,S2L1	::ALLOW ENTRY INTO SEC. 2
1895	024636	005237	016720			INC	S2LOG2	::LOG OUT OF SECTION 2 INITIALIZATION
1896	024642	023737	016774	016720	1\$:	CMP	CPUACT,S2LOG2	::WAIT FOR ALL THE CPUS TO GET HERE
1897	024650	001374				BNE	1\$	
1898	024652	012777	040602	172154		MOV	#ISTDIS,@ISTVEC	::POINT TO DISPATCHER
1899	024660	000137	024674			JMP	TST17	::START THE TEST
1900								
1901	024664	000240			NORP:	NOP		::THERE IS NO RP CONTROLLER
1902	024666	012716	024516			MOV	#NORH70,(SP)	::SET FOR TEST ENTRY RETURN

1903 024672 000002

RTI

:::RETURN

1904
1905
1906
1907
1908
1909
1910
1911
1912

:::*****
:*TEST 17 CHECK 'BRK' & 'DCF' FLAGS DURING POWERFAIL
:*****

1913
1914
1915

TST17:

1916 024674
1917 024674 012777 000017 167266

MOV #17,@DISPLAY ;SET TEST NUMBER

1918
1919

:::***** TS17A-FORK *****

1920

1921 024702 016006 014200
1922 024706 052777 100000 172114

MOV \$\$STP(R0), SP;SET UP THE STACK
BIS #BIT15,@ACR ;INITIALIZE THE IIST
ASRB @69\$;ENTER CONTROL FORK

1923 024714 106277 000126
1924 024720 103375
1925 024722 027737 000114 016774

70\$:

BCC 70\$
CMP @67\$,CPUACT ;HAVE WE REACHED THE END OF THE
;ROUTING CYCLE?

1926

BNE 65\$;BRANCH IF NO

1927 024730 001021
1928 024732 013702 016774

MOV CPUACT,R2

1929 024736 005302
1930 024740 006302

DEC R2

1931 024742 027702 000076

ASL R2 ;(CPUACT-1)*2
CMP @68\$,R2 ;ARE WE AT THE END OF THE TEST?

1932 024746 001005
1933 024750 012777 000001 000070

BNE 64\$;BRANCH IF NO

1934 024756 000137 025524
1935 024762 062777 000002 000054 64\$:

MOV #1,@69\$;EXIT
JMP TST20

1936 024770 005077 000046
1937 024774 005277 000042 65\$:

ADD #2,@68\$;INCREMENT 68\$ BY 2
CLR @67\$;CLEAR THE CHECKPOINT COUNTER

1938 025000 005037 014642
1939 025004 005037 014714

INC @67\$;INCREMENT CHECKPOINT

1940 025010 020077 000030
1941 025014 001005

CLR SIGNAL ;CLEAR SIGNAL LOCK
CLR SYNC.2 ;CLEAR THE LOCK

1942 025016 012777 000001 000022
1943 025024 000137 025050

CMP R0,@68\$;ROUTE THIS PROCESSOR THROUGH TS17A?
BNE 66\$;BRANCH IF NO

1944 025030 012777 000001 000010 66\$:
1945 025036 000137 025172

MOV #1,@69\$;CLEAR LOCK
JMP TS17A ;JUMP TO BRANCH TS17A

1946
1947 025042 016722 67\$:
1948 025044 016724 68\$:
1949 025046 017004 69\$:

JMP TS17B ;JUMP TO TS17B

1950

:::*****

1951 025050 112761 000017 014002 TS17A:

MOVB #17,\$TSTNM(R1) ;SET UP THE TEST NUMBER

1952 025056 010037 017062

MOV R0,PUT ;SET PROCESSOR UNDER TEST

1953 025062 005700
1954 025064 001007

TST R0 ;IS THIS THE MASTER?
BNE 5\$;BRANCH IF NO

1955 025066 104401 041713
1956 025072 005046

TYPE ,TM77 ;'TEST'
CLR -(SP)

1957 025074 116116 014002
1958 025100 104403

MOVB \$TSTNM(R1),(SP) ;GET THE TEST NO.
TYPOS

```

1959 025102 000002          .WORD 2          ;TYPE 2 DIGITS, NO LEADING 0
1960 025104          5$:
1961 025104 104401 041724      TYPE          TM100          ;'POWER FAIL CPU #'
1962 025110 016046 014226      MOV          $CPUID(R0),-(SP) ;:SAVE $CPUID(R0) FOR TYPEOUT
1963 025114 104405          TYPDS          ;:GO TYPE--DECIMAL ASCII WITH SIGN
1964 025116 104401 014333      TYPE          ,SCLRF
1965 025122 012760 014020 014612  MOV          #SSU!TI!NCX,PFPT(R0) ;SEND SIG. ON UP, TIME, DON'T SAVE MM
1966 025130 012760 025160 014672  MOV          #BAD,ISTTAB(R0) ;SET UP IIST VECTOR FOR THIS CPU.
1967 025136 005003          CLR          R3          ;RETURN AFTER THE WAIT ON POWER UP
1968 025140 005037 014642          CLR          SIGNAL      ;CLEAR THE POWER UP SIGNAL
1969 025144 012737 000001 014714  MOV          #1, SYNC.2    ;UNLOCK THE OTHER CPUS
1970 025152 000001          WAIT          ;WAIT FOR THE POWER TO FAIL.
1971 025154 000137 024674          JMP          TST17        ;GO TO CONTROL FORK
1972
1973
1974 025160 104025          BAD:          ERROR          25          ;UNEXPECTED CPU INTERRUPT
1975 025162 000000          HALT
1976 025164 012716 024674      MOV          #TST17, (SP)   ;CONTINUE TESTING
1977 025170 000002          RTI
1978
1979 025172 112761 000017 014002  TS17B:      MOVB          #17, $STSTM(R1) ;SET UP THE TEST NUMBER
1980 025200 106237 014714          ASRB          SYNC.2      ;WAIT FOR P.U.T. TO SEND SIGNAL
1981 025204 103372          BCC          1S17B        ;LOOP UNTIL SENT
1982 025206 005237 016764          INC          LOG1         ;COUNT CPU'S COMING THRU
1983 025212 023737 016764 016770  CMP          LOG1,CPULS1   ;ALLHERE?
1984 025220 001404          BEQ          2$          ;YES - SKIP
1985 025222 012737 000001 014714  MOV          #1,SYNC.2     ;NO, LET NEXT ONE IN
1986 025230 000402          BR          5$          ;THEN CONTINUE
1987 025232 005037 016764          CLR          LOG1         ;LAST ONE INITIALIZES COUNTER
1988 025236 104401 014220          5$:          TYPE          ,SNUL          ;FLUSH THE TYPE QUEUE
1989 025242 012760 025160 014672  MOV          #BAD,ISTTAB(R0) ;GET SET FOR BAD INTERRUPT
1990 025250 012777 000001 171552  MOV          #PGCS,@ACR    ;ACCESS PGCS REGISTER
1991 025256 052777 000004 171546  BIS          #BIT2,@ADR    ;SET THE INTERRUPT ENABLE BIT
1992
1993 025264 012777 000007 171536          MOV          #DCF,@ACR    ;COPY THE DCF REG.
1994 025272 017705 171534          MOV          @ADR,R5      ;INTO R5
1995 025276 012760 025376 014672  MOV          #STS17,ISTTAB(R0) ;SET FOR EXPECTED INTERRUPT
1996 025304 106237 014642          1$:          ASRB          SIGNAL      ;WAIT ON POWER FAIL SIGNAL
1997 025310 103375          BCC          1$
1998 025312 012737 000001 014642  MOV          #1,SIGNAL     ;LET ANOTHER IN
1999 025320 017760 171504 014250  MOV          @ACR,$REG1(R0) ;SAVE THE ACR
2000 025326 012777 000000 171474  MOV          #PGTE,@ACR    ;ACCESS THE PGTE REG.
2001 025334 017760 171472 014260  MOV          @ADR,$REG2(R0) ;SAVE THE PGTE REG.
2002 025342 017760 171464 014270  MOV          @ADR,$REG3(R0) ;SAVE THE PGCS REG.
2003 025350 104020          ERROR          20          ;NO IIST INTERRUPT
2004 025352 005237 016766          INC          LOG2         ;COUNT CPU'S GOING OUT
2005 025356 023737 016766 016770  CMP          LOG2,CPULS1   ;ALL DONE?
2006 025364 001002          BNE          10$         ;NO - SKIP
2007 025366 005037 016766          CLR          LOG2         ;YES, INIT FOR NEXT CYCLE
2008 025372 000137 024674          10$:        JMP          TST17        ;DO IT AGAIN
2009 025376          STS17:
2010 025376 012777 000007 171424  MOV          #DCF,@ACR    ;ACCESS DCF REGISTER
2011 025404 013703 017062          MOV          PUT,R3      ;GET LOGICAL ID INTO R3
2012 025410 016304 014226          MOV          $CPUID(R3),R4 ;COPY IIST ID TO DESTINATION.
2013 025414 012703 000401          MOV          #401,R3     ;MAKE A MASK
2014 025420 072304          ASH          R4,R3      ;BRING IT INTO POSITION

```

N 4

```

2015 025422 050305      BIS      R3,R5      ;R5 IS WHAT THE DCF REG. SHOULD LOOK LIKE
2016 025424 077301      SOB      R3          ;DELAY A SHORT WHILE
2017 025426 017702 171400    MOV      @ADR,R2 ;COPY DCF REGISTER
2018 025432 020205      CMP      R2,R5      ;EVERYTHING OK?
2019 025434 001405      BEQ      3$         ;BRANCH IF YES
2020 025436 010260 014250    MOV      R2,$REG1(R0) ;THE DCF REG.
2021 025442 010560 014260    MOV      R5,$REG2(R0) ;WHAT IT SHOULD BE
2022 025446 104021      ERROR   21         ;INCORRECT DCF REG. BITS
2023 025450 005060 017012    3$:     CLR      NOPRMP(R0)
2024 025454 104401 045163    TYPE    ,TM111     ;INTERRUPT AS EXPECTED
2025 025460 106237 014712    4$:     ASRB    SYNC.1 ;WAIT FOR POWER-UP
2026 025464 103375      BCC     4$
2027 025466 005237 016766    INC     LOG2        ;COUNT CPU'S GOING OUT
2028 025472 023737 016766 016770    CMP     LOG2,CPULS1 ;ALL DONE?
2029 025500 001404      BEQ     6$         ;YES - SKIP
2030 025502 012737 000001 014712    MOV     #1,SYNC.1  ;ELSE LLET NEXT ONE IN
2031 025510 000402      BR     7$
2032 025512 005037 016766    6$:     CLR     LOG2        ;LAST ONE INITS COUNTER
2033 025516 012716 024674    7$:     MOV     #TST17, (SP) ;CONTINUE
2034 025522 000002      RTI
2035
2036
2037
2038
2039
2040
2041
2042 025524
2043 025524 012777 000020 166436    TST20: MOV     #20,@DISPLAY ;SET TEST NUMBER
2044
2045
2046 025532 016006 014200    MOV     $$STP(R0), SP ;SET UP THE STACK
2047 025536 052777 100000 171264    70$:   BIS     #BIT15, @ACR ;INITIALIZE THE IIST
2048 025544 106277 000122      ASRB    @69$        ;ENTER CONTROL FORK
2049 025550 103375      BCC     70$
2050 025552 027757 000110 016774    CMP     @67$,CPUACT ;HAVE WE REACHED THE END OF THE
2051
2052 025560 001021      BNE     65$        ;ROUTING CYCLE?
2053 025562 013702 016774      MOV     CPUACT,R2  ;BRANCH IF NO
2054 025566 005302      DEC     R2
2055 025570 006302      ASL     R2          ;(CPUACT-1)*2
2056 025572 027702 000072    CMP     @68$,R2    ;ARE WE AT THE END OF THE TEST?
2057 025576 001005      BNE     64$        ;BRANCH IF NO
2058 025600 012777 000001 000064    MOV     #1,@69$    ;EXIT
2059 025606 000137 026114      JMP     TST21
2060 025612 062777 000002 000050    64$:   ADD     #2,@68$   ;INCREMENT 68$ BY 2
2061 025620 005077 000042      CLR     @67$        ;CLEAR THE CHECKPOINT COUNTER
2062 025624 005277 000036    65$:   INC     @67$        ;INCREMENT CHECKPOINT
2063 025630 005037 014714      CLR     SYNC.2     ;CLEAR THE LOCK
2064 025634 020077 000030    CMP     R0,@68$    ;ROUTE THIS PROCESSOR THROUGH TS20A?
2065 025640 001005      BNE     66$        ;BRANCH IF NO
2066 025642 012777 000001 000022    MOV     #1,@69$    ;CLEAR LOCK
2067 025650 000137 025674      JMP     TS20A      ;JUMP TO BRANCH TS20A
2068 025654 012777 000001 000010    66$:   MOV     #1,@69$    ;CLEAR LOCK
2069 025662 000137 026006      JMP     TS20B      ;JUMP TO TS20B
2070

```

 ;*TEST 20 CHECK POWERFAIL DURING HIGH MEMORY ACTIVITY

TST20:
 ;***** TS20A-FORK *****

 ;***** TS20B *****

```

2071 025666 016726 67$: D1
2072 025670 016730 68$: D2
2073 025672 017006 69$: D3
2074 025674 112761 000020 014002 TS20A: MOV B #20, $STSTM(R1)
2075 025702 010037 017062 MOV RO,PUT ;SET PROCESSOR UNDER TEST
2076 025706 005700 TST RO ;IS THIS THE MASTER?
2077 025710 001007 BNE 5$ ;BRANCH IF NO
2078 025712 104401 041713 TYPE .TM77 ;'TEST'
2079 025716 005046 CLR -(SP)
2080 025720 116116 014002 MOV B $STSTM(R1),(SP) ;GET THE TEST NO.
2081 025724 104403 TYPOS
2082 025726 000002 .WORD 2 ;TYPE 2 DIGITS, NO LEADING 0
2083 025730 5$:
2084 025730 104401 041724 TYPE .TM100 ;'POWERFAIL CPU # ''
2085 025734 016046 014226 MOV $CPUID(RO),-(SP) ;SAVE $CPUID(RO) FOR TYPEOUT
2086 025740 104405 TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2087 025742 104401 014333 TYPE .$RCLF
2088 025746 012760 014020 014612 MOV #SSU!TI!NCX,PFRT(RO) ;SEND SIGNAL ON UP,TIME,DON'T SAVE NV
2089 025754 005003 CLR R3 ;SET FOR RTI RETURN
2090 025756 023737 016726 016774 1$: CMP D1,CPUACT ;MAKE SURE ALL CPU'S DONE WITH
2091 025764 001374 BNE 1$ ;PREVIOUS CYCLE
2092 025766 005037 014642 CLR SIGNAL ;CLEAR THE POWER-UP SIGNAL
2093 025772 012737 000001 014714 MOV #1, SYNC.2 ;UNLOCK THE OTHER CPUS
2094 026000 000001 WAIT ;WAIT FOR THE POWER TO FAIL
2095 026002 000137 025524 JMP TST20
2096
2097 026006 112761 000020 014002 TS20B: MOV B #20, $STSTM(R1);SET UP THE TEST NUMBER
2098 026014 106237 014714 ASRB SYNC.2 ;WAIT FOR SYNC. SIGNAL
2099 026020 103372 BCC TS20B
2100 026022 005237 016764 INC LOG1 ;COUNT CPU'S GOING THRU
2101 026026 023737 016764 016770 CMP LOG1,CPU1S1 ;LAST ONE HERE?
2102 026034 001404 BEQ 3$ ;YES - SKIP
2103 026036 012737 000001 014714 MOV #1,SYNC.2 ;ELSE LET NEXT ONE IN
2104 026044 000402 BR 4$
2105 026046 005037 016764 3$: CLR LOG1 ;LAST ONE INITS COUNTER
2106 026052 104401 014220 4$: TYPE $NULL ;FLUSH THE TYPE QUEUE
2107 026056 012705 002000 1$: MOV #2000,R5 ;INITIALIZE COUNTER
2108 026062 011010 2$: MOV (RO),(RO)
2109 026064 011010 MOV (RO),(RO)
2110 026066 011010 MOV (RO),(RO)
2111 026070 011010 MOV (RO),(RO)
2112 026072 077505 SOB R5,2$
2113 026074 106237 014642 ASRB SIGNAL ;SIGNAL RECEIVED?
2114 026100 103366 BCC 1$ ;NO,CONTINUE WITH CONTENSION
2115 026102 012737 000001 014642 MOV #1,SIGNAL ;OPEN LOCK FOR NEXT CPU
2116 026110 000137 025524 JMP TST20
2117
2118
2119
2120
2121
2122
2123
2124 026114 012777 000021 166046 TST21: MOV #21,@DISPLAY ;SET TEST NUMBER
2125 026114
2126

```

```

*****
*TEST 21 CHECK POWERFAIL SEQUENCE DURING MASSBUS XFER
*****
TST21:
***** TS21A-FORK *****

```

```

2127
2128 026122 016006 014200 MOV    $$STP(R0), SP;SET UP THE STACK
2129 026126 052777 100000 170674 BIS    #BIT15, @ACR ;INITIALIZE THE IIST
2130 026134 106277 000126 70$: ASRB  @69$ ;ENTER CONTROL FORK
2131 026140 103375 BCC   70$
2132 026142 027737 000114 016774 CMP    @67$,CPUACT ;HAVE WE REACHED THE END OF THE
2133                                     ;ROUTING CYCLE?
2134 026150 001021 BNE   65$ ;BRANCH IF NO
2135 026152 013702 016774 MOV    CPUACT,R2
2136 026156 005302 DEC    R2
2137 026160 006302 ASL   R2 ;(CPUACT-1)*2
2138 026162 027702 000076 CMP    @68$,R2 ;ARE WE AT THE END OF THE TEST?
2139 026166 001005 BNE   64$ ;BRANCH IF NO
2140 026170 012777 000001 000070 MOV    #1, @69$ ;EXIT
2141 026176 000137 027076 JMP    TST22
2142 026202 062777 000002 000054 64$: ADD   #2, @68$ ;INCREMENT 68$ BY 2
2143 026210 005077 000046 CLR    @67$ ;CLEAR THE CHECKPOINT COUNTER
2144 026214 005277 000042 65$: INC   @67$ ;INCREMENT CHECKPOINT
2145 026220 005037 014712 CLR    SYNC.1 ;CLEAR ALL THE LOCKS
2146 026224 005037 014714 CLR    SYNC.2 ;CLEAR THE LOCK
2147 026230 020077 000030 CMP    R0,@68$ ;ROUTE THIS PROCESSOR THROUGH TS21A?
2148 026234 001005 BNE   66$ ;BRANCH IF NO
2149 026236 012777 000001 000022 MOV    #1,@69$ ;CLEAR LOCK
2150 026244 000137 026270 JMP    TS21A ;JUMP TO BRANCH TS21A
2151 026250 012777 000001 000010 66$: MOV    #1,@69$ ;CLEAR LOCK
2152 026256 000137 026656 JMP    TS21B ;JUMP TO TS21B
2153                                     ;*****
2154 026262 016732 67$: E1
2155 026264 016734 68$: E2
2156 026266 017010 69$: E3
2157 026270 112761 000021 014002 TS21A: MOVB  #21, $TSTNM(R1)
2158 026276 010037 017062 MOV    R0,PUT ;SET PROCESSOR UNDER TEST
2159 026302 023737 016732 016774 4$: CMP    E1, CPUACT ;LET THE OTHER CPUS CATCH UP
2160 026310 001374 BNE   4$
2161 026312 005700 TST   R0 ;IS THIS THE MASTER?
2162 026314 001007 BNE   5$ ;BRANCH IF NO
2163 026316 104401 041713 TYPE  ,TM77 ;'TEST'
2164 026322 005046 CLR    -(SP)
2165 026324 116116 014002 MOVB  $TSTNM(R1),-(SP) ;GET THE TEST NO.
2166 026330 104403 TYPOS
2167 026332 000002 .WORD  2 ;TYPE 2 DIGITS, NO LEADING 0
2168 026334 5$:
2169 026334 013702 016774 MOV    CPUACT, R2 ;CHECK FOR MASSBUS DEVICES ON OTHER CPUS
2170 026340 006302 ASL   R2
2171 026342 005005 CLR    R5
2172 026344 020205 1$: CMP    R2, R5
2173 026346 001407 BEQ   2$
2174 026350 022765 001400 014632 CMP    #1400, MBDSW(R5)
2175 026356 001020 BNE   3$
2176 026360 062705 000002 ADD    #2, R5
2177
2178 026364 000767 BR    1$
2179 026366 104401 044147 2$: TYPE  ,NODEV ;THERE ARE NO DEVICES TO TEST THIS CPU
2180 026372 016046 014226 MOV    $CPUID(R0),-(SP) ;SAVE $CPUID(R0) FOR TYPEOUT
2181 026376 104405 TYPDS ;GO TYPE--DECIMAL ASCII WITH SIGN
2182 026400 104401 014333 TYPE  ,$CRLF
    
```


2183	026404	104401	042021		TYPE	,TM102	:'PROCEEDING TO NEXT CPU'
2184	026410	012737	000001	014714	MOV	#1, SYNC.2	
2185	026416	000636			BR	TST21	:BRANCH TO START OF TEST
2186	026420	020005			3\$: CMP	R0, R5	
2187	026422	001003			BNE	P21	:BRANCH IF THERE IS A DEVICE ON ANOTHER CPU
2188	026424	062705	000002		ADD	#2, R5	:THE DEVICE IS ON THIS CPU -
2189	026430	000745			BR	1\$	
2190	026432	104401	041724		P21: TYPE	,TM100	:'POWER FAIL CPU #'
2191	026436	016046	014226		MOV	\$CPUID(R0),-(SP)	::SAVE \$CPUID(R0) FOR TYPEOUT
2192	026442	104405			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
2193	026444	104401	014333		TYPE	,\$CRLF	
2194	026450	012760	014020	014612	MOV	#SSU!TI.NCX,PFFT(R0)	:SEND SIGNAL ON UP TIME, DON'T SAVE NW
2195	026456	012737	000001	014712	MOV	#1, SYNC.1	:UNLOCK THE OTHER CPU'S
2196	026464	005037	014642		CLR	SIGNAL	:CLEAR THE POWER-UP SIGNAL
2197	026470	005003			CLR	R3	:COME UP VIA RTI
2198	026472	022760	001400	014632	CMP	#1400,MBDSW(R0)	:DOES THIS CPU HAVE A MASSBUS DEVICE?
2199	026500	001006			BNE	1\$:BRANCH IF YES
2200	026502	000001			WAIT		:WAIT FOR POWER TO FAIL
2201	026504	012737	000001	014714	MOV	#1, SYNC.2	:UNLCK CPUS IF ANY ARE LOCKED
2202	026512	000137	026114		JMP	TST21	:GO TO CONTROL FORK
2203	026516	016004	017042		1\$: MOV	BFADR(R0),R4	:PUT ADDRESS OF BUFFER IN R4
2204	026522	012737	000070	176700	MOV	#70,@RPCS1	:DO A READ
2205	026530	004737	033540		JSR	PC,MBUSR	:READ A RECORD
2206	026534	005060	014650		CLR	CKSUM(R0)	:CLEAR CHECKSUM LOCATION
2207	026540	012702	004000		MOV	#4000,R2	:INITIALIZE A COUNTER
2208	026544	016004	017042		MOV	BFADR(R0),R4	:GET BUFFER POINTER
2209	026550	062460	014650		2\$: ADD	(R4)+,CKSUM(R0)	:PERFORM
2210	026554	005560	014650		ADC	CKSUM(R0)	:CHECKSUM
2211	026560	077205			SOB	R2,2\$:LOOP
2212	026562	016004	017042		4\$: MOV	BFADR(R0),R4	:LOAD BUFFER ADDRESS
2213	026566	012737	000050	176700	MOV	#50,@RPCS1	:DO A WRITE CHECK
2214	026574	004737	033540		JSR	PC,MBUSR	:READ FROM MASS BUS DEVICE
2215							
2216	026600	005005			CLR	R5	:CLEAR R5
2217	026602	012702	004000		MOV	#4000,R2	:INITIALIZE COUNTER
2218	026606	016004	017042		MOV	BFADR(R0),R4	:GET POINTER TO BUFFER
2219	026612	062405			5\$: ADD	(R4)+,R5	:PERFORM
2220	026614	005505			ADC	R5	:CHECKSUM
2221	026616	077203			SOB	R2,5\$	
2222	026620	020560	014650		CMP	R5,CKSUM(R0)	:EVERYTHING OK?
2223	026624	001401			BEQ	6\$:BRANCH IF YES
2224	026626	104023			ERROR	23	:CHECKSUM IS WRONG
2225	026630	106237	014642		6\$: ASRB	SIGNAL	
2226	026634	103352			BCC	4\$:NO CONTINUE XFERS
2227	026636	012737	000001	014642	MOV	#1,SIGNAL	:OPEN LOCK FOR OTHER CPU'S
2228	026644	012737	000001	014714	MOV	#1,SYNC.2	:UNLCK CPUS IF ANY ARE LOCKED
2229	026652	000137	026114		JMP	TST21	:GO TO CONTROL FORK
2230	026656	112761	000021	014002	TS21B: MOV	#21,\$STSTM(R1)	:SET THE TEST NUMBER
2231	026664	022760	001400	014632	CMP	#1400, MBDSW(R0)	:DOES THIS CPU HAVE A MASSBUS DEVICE?
2232	026672	001012			BNE	1\$:BRANCH IF YES
2233	026674	106237	014714		10\$: ASRB	SYNC.2	:ELSE SET OUT THIS ROUND
2234	026700	103375			BCC	10\$	
2235	026702	012737	000001	014714	MOV	#1,SYNC.2	:OPEN LOCK FOR OTHERS
2236	026710	104401	014220		TYPE	, \$NULL	:FLUSH THE TYPE QUEUE
2237	026714	000137	026114		JMP	TST21	:JUMP INTO THE CONTROL LOOP
2238	026720				1\$:		


```

2239 026720 016004 017042      MOV      BFADR(R0),R4      ;PUT ADDRESS OF BUFFER IN R4
2240 026724 012737 000070 176700  MOV      #70,@#RPCS1      ;DO A READ
2241 026732 004737 033540      JSR      PC,MBUSR         ;READ A RECORD
2242 026736 005060 014650      CLR      CKSUM(R0)        ;CLEAR CHECKSUM LOCATION
2243 026742 012703 004000      MOV      #4000,R3         ;INITIALIZE A COUNTER
2244 026746 016004 017042      MOV      BFADR(R0),R4      ;GET BUFFER POINTER
2245 026752 062460 014650 2$:  ADD      (R4)+,CKSUM(R0)   ;PERFORM...
2246 026756 005560 014650      ADC      CKSUM(R0)        ;CHECKSUM.
2247 026762 077305              SOB      R3,2$           ;LOOP
2248 026764 106237 014712 7$:  ASRB     SYNC.1           ;HOLD UP
2249 026770 103375              BCC      7$
2250 026772 012737 000001 014712  MOV      #1,SYNC.1        ;OPEN LOCK FOR OTHERS
2251 027000 104401 014220      TYPE     ,SNULL           ;FLUSH THE QUEUE
2252 027004 016004 017042 4$:  MOV      BFADR(R0),R4      ;LOAD BUFFER ADDRESS
2253 027010 012737 000050 17670C  MOV      #50,@#RPCS1      ;DO A WRITE CHECK
2254 027016 004737 033540      JSR      PC,MBUSR         ;READ FROM MASS BUS DEVICE
2255 027022 005005              CLR      R5               ;CLEAR R5
2256 027024 012703 004000      MOV      #4000,R3         ;INITIALIZE COUNTER
2257 027030 016004 017042      MOV      BFADR(R0),R4      ;GET POINTER TO BUFFER
2258 027034 062405 5$:  ADD      (R4)+,R5         ;PERFORM...
2259 027036 005505              ADC      R5               ;CHECKSUM
2260 027040 077303              SOB      R3,5$
2261 027042 020560 014650      CMP      R5,CKSUM(R0)     ;EVERYTHING OK?
2262 027046 001401              BEQ      6$               ;BRANCH IF YES
2263 027050 104023              ERROR    23               ;CHECKSUM IS WRONG
2264 027052 106237 014642 6$:  ASRB     SIGNAL           ;NO CONTINUE XFERS
2265 027056 103352              BCC      4$               ;OPEN LOCK FOR OTHERS
2266 027060 012737 000001 014642  MOV      #1,SIGNAL
2267 027066 104401 014220      TYPE     ,SNULL
2268 027072 000137 026114      JMP      TST21            ;JUMP INTO CONTROL LOOP
  
```

```

2269
2270
2271
2272
2273
2274
2275
2276
2277
2278      ;*****
2279      ;*TEST 22      CHECK AC POWERFAIL ON MEM BOXES, PORTS DISABLED
2280      ;*****
2281      TST22:
2282      MOV      #22,@DISPLAY ;SET TEST NUMBER
2283      MOV      #22,@STSTM(R1) ;SET THE TEST NUMBER
2284      ASRB     SYNC.3        ;CONTROL THE ENTRY
2285      BCC      2$
2286      INC      ENTR22        ;INCREMENT ENTER FLAG
2287      MOV      #1,SYNC.3    ;ALLOW THE OTHERS IN
2288      CMP      CPUACT, ENTR22 ;ARE ALL CPUS HERE?
2289      BNE      1$           ;NOT YET
2290      CLR      EXIT         ;CLEAR THE EXIT FLAG
2291      CLR      PSW         ;SET KERNAL MODE
2292      CMP      R0, #0       ;IS THIS THE MASTER?
2293      BNE      TS22B        ;BRANCH IF NO
2294      BR      TS22A         ;THIS IS THE MASTER
  
```

```

2295
2296 027162 052777 100000 167640 TS22B: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2297 027170 012777 000022 164772 MOV #22, @DISPLAY ;SET TEST NUMBER
2298 027176 016006 014200 MOV $$STP(R0), SP ;INITIALIZE THE STACK
2299 027202 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
2300 027206 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
2301 027214 012703 027326 MOV #100$, R3 ;SET FOR POWER FAIL RETURN
2302 027220 005737 014540 TST EXIT ;FINISHED WITH THIS TEST?
2303 027224 001404 BEQ 1$ ;BRANCH IF NO
2304 027226 005037 177572 CLR MMRO ;MAKE SURE MM IS TURNED OFF
2305 027232 000137 030120 JMP TST23 ;GO TO NEXT TEST
2306 027236 005737 014534 1$: TST RELOUP ;TIME TO RELOCATE?
2307 027242 001421 BEQ 2$ ;BRANCH IF NO
2308 027244 004737 032604 JSR PC, SETMM ;GET READY FOR RELOCATION
2309 027250 063737 014550 172340 ADD HIBOX, KIPAR0
2310 027256 063737 014550 172342 ADU HIBOX, KIPAR1
2311 027264 063737 014550 172344 ADD HIBOX, KIPAR2
2312 027272 052737 000001 177572 BIS #1, MMRO ;SLAVE IS NOW IN HIGH CORE
2313 027300 005037 014534 CLR RELOUP ;CLEAR RELOCATION FLAG
2314 027304 000726 BR TS22B ;CONTINUE TESTING
2315 027306 005737 014536 2$: TST RELODN ;TIME TO RELOCATE?
2316 027312 001723 BEQ TS22B ;BRANCH IF NO
2317 027314 005037 177572 CLR MMRO ;RETURN TO LOW CORE
2318 027320 005037 014536 CLR RELODN ;CLEAR THE FLAG
2319 027324 000716 BR TS22B ;CONTINUE
2320
2321 027326 005737 014522 100$: TST PWRFL ;SHOULD WE BE HERE?
2322 027332 001002 BNE 101$ ;BRANCH IF YES
2323 027334 104001 ERROR 1 ;UNEXPECTED CPU POWER FAIL
2324 027336 000711 BR TS22B ;CONTINUE TESTING
2325 027340 005060 017012 101$: CLR NOPRMP(R0) ;WANT TO IDENTIFY THE CPU
2326 027344 104401 042052 TYPE ,TM103 ;EXPECTED CPU POWER FAIL
2327 027350 000704 BR TS22B ;CONTINUE TESTING
2328
2329
2330
2331 027352 TS22A:
2332 027352 005700 TST R0 ;IS THIS THE MASTER?
2333 027354 001007 BNE 5$ ;BRANCH IF NO
2334 027356 104401 041713 TYPE ,TM77 ;'TEST'
2335 027362 005046 CLR -(SP)
2336 027364 116116 014002 MOV $TSTNM(R1), (SP) ;GET THE TEST NO.
2337 027370 104403 TYPOS
2338 027372 000002 .WORD 2 ;TYPE 2 DIGITS, NO LEADING 0
2339 027374
2340 027374 012737 000000 014522 5$: MOV #0, PWRFL ;SPECIFY WHETHER OR NOT TO EXPECT CPU POWER FAIL
2341 027402 012737 000000 014526 MOV #0, BOOT ;SPECIFY WHETHER OR NOT TO EXPECT CPU BOOT AND I
2342 027410 004737 032342 JSR PC, MEMSIZ ;FIND ALL THE MEM BOXES
2343 027414 005737 014540 TST EXIT ;WAS ONLY ONE MEM BOX FOUND?
2344 027420 001402 BEQ 1$ ;BRANCH IF NO
2345 027422 000137 030120 JMP TST23 ;WE CAN'T DO THIS TEST
2346 027426 012702 000016 1$: MOV #16, R2 ;POINT TO BOX #7
2347 027432 052777 100000 167370 2$: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2348 027440 012777 000022 164522 MOV #22, @DISPLAY ;SET THE TEST NUMBER
2349 027446 016006 014200 MOV $$STP(R0), SP ;INITIALIZE THE STACK
2350 027452 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
    
```

```

2351 027456 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
2352 027464 005702 TST R2 ;DID WE TEST ALL THE BOXES
2353 027466 002012 BGE 3$ ;BRANCH IF NO
2354 027470 012737 000112 000110 MOV #112, @#110 ;RESTORE LOC 110
2355 027476 005037 000000 CLR @#0 ;RESTORE LOC 0
2356 027502 012737 000001 014540 MOV #1, EXIT ;SIGNAL THE SLAVES TO EXIT
2357 027510 000137 030120 JMP TST23 ;GO TO THE NEXT TEST
2358 027514 005762 014502 3$: TST START(R2) ;WHAT DO WE KNOW ABOUT THIS BOX?
2359 027520 003037 BGT 10$ ;BRANCH IF NOT THE BASE BOX
2360 027522 001403 BEQ 4$ ;BRANCH IF ITS THE BASE BOX
2361 027524 162702 000002 SUB #2, R2 ;THERE WAS NO BOX-POINT TO THE NEXT LOWER BOX
2362 027530 000740 BR 2$ ;CONTINUE
2363 027532 022737 000001 014646 4$: CMP #1, BOXNUM ;WAS THERE ONLY ONE MEM BOX?
2364 027540 002027 BGE 11$ ;BRANCH IF YES
2365 027542 005737 014526 TST BOOT ;IS THIS THE DC TEST?
2366 027546 001141 BNE 103$ ;BRANCH IF YES
2367 027550 004737 033042 JSR PC, RELOHI ;TO TEST THE BASE BOX,
2368 ;RELOCATE THE PROGRAM FIRST TO THE NEXT HIGHER B
2369 027554 012737 000001 014532 MOV #1, HICORE ;TURN MM ON ON POWER-UP
2370 027562 012737 000001 014534 MOV #1, RELOUP ;SIGNAL SLAVES TO RELOCATE
2371 027570 063737 014550 172340 ADD HIBOX, KIPAR0 ;GET READY TO GO TO HIGH CORE
2372 027576 063737 014550 172342 ADD HIBOX, KIPAR1
2373 027604 063737 014550 172344 ADD HIBOX, KIPAR2
2374 027612 052737 000001 177572 BIS #1, MMRO ;WE ARE NOW IN HIGH CORE
2375 027620 10$:
2376 027620 005037 014530 11$: CLR PATCHK ;SET TO WRITE PATTERN
2377 027624 004737 032700 JSR PC, PATTRN ;WRITE THE PATTERN
2378 027630 004737 032566 JSR PC, BUFCLR ;CLEAR THE KEYBOARD BUFFER
2379 027634 104401 042534 TYPE ,TM106 ;TELL THE OPERATOR TO POWER FAIL THE MEMORY BOX
2380 027640 006202 ASR R2
2381 027642 010246 MOV R2, -(SP)
2382 027644 104405 TYPDS
2383 027646 006302 ASL R2
2384 027650 104401 043226 TYPE ,TM108 ;SPECIFY THE CONDITIONS
2385 027654 012703 030016 MOV #100$, R3 ;SET UP THE POWER FAIL RETURN
2386 027660 005737 014526 TST BOOT ;IS THIS THE DC TEST?
2387 027664 001414 BEQ 12$ ;BRANCH IF NO
2388 027666 012737 020000 000000 13$: MOV #20000, @#0 ;MAKE BOOTING SLAVE SP=20000
2389 027674 005237 000110 INC @#110 ;HANG THE SLAVES BOOT
2390 027700 105777 164304 TSTB @STKS ;IS CHARACTER IN BUFFER?
2391 027704 100370 BPL 13$ ;LOOP
2392 027706 012777 033226 167120 MOV #ENTR, @ISTVEC ;SET UP TO INTERRUPT SLAVES
2393 027714 000413 BR 14$ ;CONTINUE
2394 027716 12$:
2395 027716 105777 164266 TSTB @STKS ;IS CHARACTER IN THE BUFFER?
2396 027722 100375 BPL 12$ ;LOOP
2397 027724 005737 014522 17$: TST PWRFL ;SHOULD THE MASTER HAVE POWER FAILED?
2398 027730 001402 BEQ 18$ ;BRANCH IF NO
2399 027732 104024 ERROR 24 ;FAILURE TO POWER FAIL
2400 027734 000441 BR 102$ ;CONTINUE
2401 027736 104401 047252 18$: TYPE ,OK
2402 027742 000436 BR 102$
2403 027744 017704 164210 14$: MOV @SWR, R4 ;GET THE SWITCH VALUES
2404 027750 042704 177760 BIC #177760, R4 ;SAVE ONLY CPU BITS
2405 027754 012777 000000 167046 MOV #PGTE, @ACR ;ACCESS PGTE REG
2406 027762 010477 167044 MOV R4, @ADR ;SET INTERRUPT BITS

```

```

2407 027766 032737 000001 016774 BIT #BIT0, CPUACT ;EVEN OR ODD?
2408 027774 001404 BEQ 15$ ;BRANCH IF EVEN
2409 027776 012777 000001 167026 MOV #1, @ADR ;INTERRUPTING AN EVEN NUMBER OF SLAVES
2410 030004 000422 BR 103$ ;
2411 030006 012777 000003 167016 15$: MOV #3, @ADR ;INTERRUPTING AN ODD # OF SLAVES
2412 030014 000416 BR 103$ ;
2413 ;
2414 030016 005737 014522 100$: TST PWRFL ;DID WE EXPECT POWER FAIL?
2415 030022 001002 BNE 101$ ;BRANCH IF YES
2416 030024 104001 ERROR 1 ;MASTER ERRONEOUSLY POWER FAILED
2417 030026 000404 BR 102$ ;CONTINUE
2418 030030 005060 017012 101$: CLR NOPRMP(R0) ;ALLOW CPU IDENTIFICATION
2419 030034 104401 042052 TYPE ,TM103 ;CPU POWER FAIL MSG
2420 030040 012737 000001 014530 102$: MOV #1, PATCHK ;SET UP FOR PATTERN CHECK
2421 030046 004737 032700 JSR PC, PATTRN ;CHECK THE PATTERN
2422 030052 032737 000001 177572 103$: BIT #1, MMRO ;ARE WE IN HIGH CORE?
2423 030060 001411 BEQ 104$ ;BRANCH IF NO
2424 030062 004737 033166 JSR PC, RELOLO ;ELSE RELOCATE
2425 030066 012737 000001 014536 MOV #1, RELODN ;SIGNAL THE SLAVES
2426 030074 005037 177572 CLR MMRO ;WE ARE NOW BACK DOWN IN LOW CORE
2427 030100 005037 014532 CLR HICORE ;MAKE SURE MM ON POWER-UP DISABLED
2428 030104 005005 104$: CLR R5 ;MAKE TS22A DELAY
2429 030106 077501 SOB R5, ;
2430 030110 162702 000002 SUB #2, R2 ;POINT TO NEXT BOX
2431 030114 000137 027432 JMP 2$ ;CONTINUE
2432 ;
2433 ;
2434 ;
2435 ;
2436 ;
2437 ;
2438 ;
2439 ;
2440 ;
2441 ;
2442 030120 ;
2443 030120 012777 000023 164042 MOV #23,@DISPLAY ;SET TEST NUMBER
2444 ;
2445 030126 112761 000023 014002 2$: MOVB #23, $TSTNM(R1) ;SET THE TEST NUMBER
2446 030134 106237 016776 ASRB SYNC.3 ;CONTROL THE ENTRY
2447 030140 103375 BCC 2$ ;
2448 030142 005237 014544 INC ENTR23 ;INCREMENT ENTER FLAG
2449 030146 012737 000001 016776 1$: MOV #1, SYNC.3 ;ALLOW THE OTHERS IN
2450 030154 023737 016774 014544 CMP CPUACT, ENTR23 ;ARE ALL CPUS HERE?
2451 030162 001374 BNE 1$ ;NOT YET
2452 030164 005037 014540 CLR EXIT ;CLEAR THE EXIT FLAG
2453 030170 005037 177776 CLR PSW ;SET KERNAL MODE
2454 030174 020027 000000 CMP R0, #0 ;IS THIS THE MASTER?
2455 030200 001001 BNE TS23B ;BRANCH IF NO
2456 030202 000474 BR TS23A ;THIS IS THE MASTER
2457 ;
2458 030204 052777 100000 166616 TS23B: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2459 030212 012777 000023 163750 MOV #23, @DISPLAY ;SET TEST NUMBER
2460 030220 016006 014200 MOV $$STP(R0), SP ;INITIALIZE THE STACK
2461 030224 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
2462 030230 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
  
```

```

*****
*TEST 23 CHECK AC POWERFAIL ON MEM BOXES, PORTS ENABLED
*****
  
```

TST23:

```

2443 030120 012777 000023 164042 MOV #23,@DISPLAY ;SET TEST NUMBER
2444 ;
2445 030126 112761 000023 014002 2$: MOVB #23, $TSTNM(R1) ;SET THE TEST NUMBER
2446 030134 106237 016776 ASRB SYNC.3 ;CONTROL THE ENTRY
2447 030140 103375 BCC 2$ ;
2448 030142 005237 014544 INC ENTR23 ;INCREMENT ENTER FLAG
2449 030146 012737 000001 016776 1$: MOV #1, SYNC.3 ;ALLOW THE OTHERS IN
2450 030154 023737 016774 014544 CMP CPUACT, ENTR23 ;ARE ALL CPUS HERE?
2451 030162 001374 BNE 1$ ;NOT YET
2452 030164 005037 014540 CLR EXIT ;CLEAR THE EXIT FLAG
2453 030170 005037 177776 CLR PSW ;SET KERNAL MODE
2454 030174 020027 000000 CMP R0, #0 ;IS THIS THE MASTER?
2455 030200 001001 BNE TS23B ;BRANCH IF NO
2456 030202 000474 BR TS23A ;THIS IS THE MASTER
2457 ;
2458 030204 052777 100000 166616 TS23B: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2459 030212 012777 000023 163750 MOV #23, @DISPLAY ;SET TEST NUMBER
2460 030220 016006 014200 MOV $$STP(R0), SP ;INITIALIZE THE STACK
2461 030224 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
2462 030230 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
  
```

```

2463 030236 012703 030350 MOV #100$, R3 ;SET FOR POWER FAIL RETURN
2464 030242 005737 014540 TST EXIT ;FINISHED WITH THIS TEST?
2465 030246 001404 BEQ 1$ ;BRANCH IF NO
2466 030250 005037 177572 CLR MMRO ;MAKE SURE MM IS TURNED OFF
2467 030254 000137 031142 JMP TST24 ;GO TO NEXT TEST
2468 030260 005737 014534 1$: TST RELOUP ;TIME TO RELOCATE?
2469 030264 001421 BEQ 2$ ;BRANCH IF NO
2470 030266 004737 032604 JSR PC, SETMM ;GET READY FOR RELOCATION
2471 030272 063737 014550 172340 ADD HIBOX, KIPARO
2472 030300 063737 014550 172342 ADD HIBOX, KIPAR1
2473 030306 063737 014550 172344 ADD HIBOX, KIPAR2
2474 030314 052737 000001 177572 BIS #1, MMRO ;SLAVE IS NOW IN HIGH CORE
2475 030322 005037 014534 CLR RELOUP ;CLEAR RELOCATION FLAG
2476 030326 000726 BR TS23B ;CONTINUE TESTING
2477 030330 005737 014536 2$: TST RELODN ;TIME TO RELOCATE?
2478 030334 001723 BEQ TS23B ;BRANCH IF NO
2479 030336 005037 177572 CLR MMRO ;RETURN TO LOW CORE
2480 030342 005037 014536 CLR RELODN ;CLEAR THE FLAG
2481 030346 000716 BR TS23B ;CONTINUE
2482
2483 030350 005737 014522 100$: TST PWRFL ;SHOULD WE BE HERE?
2484 030354 001002 BNE 101$ ;BRANCH IF YES
2485 030356 104001 ERROR 1 ;UNEXPECTED CPU POWER FAIL
2486 030360 000711 BR TS23B ;CONTINUE TESTING
2487 030362 005060 017012 101$: CLR NOPRMP(R0) ;WANT TO IDENTIFY THE CPU
2488 030366 104401 042052 TYPE ,TM103 ;EXPECTED CPU POWER FAIL
2489 030372 000704 BR TS23B ;CONTINUE TESTING
2490
2491
2492
2493 030374 TS23A:
2494 030374 005700 TST R0 ;IS THIS THE MASTER?
2495 030376 001007 BNE 5$ ;BRANCH IF NO
2496 030400 104401 041713 TYPE ,TM7 ;'TEST'
2497 030404 005046 CLR -(SP)
2498 030406 116116 014002 MOVB $TSTNM(R1), (SP) ;GET THE TEST NO.
2499 030412 104403 TYPOS
2500 030414 000002 .WORD 2 ;TYPE 2 DIGITS, NO LEADING 0
2501 030416
2502 030416 012737 000001 014522 5$: MOV #1, PWRFL ;SPECIFY WHETHER OR NOT TO EXPECT CPU POWER FAIL
2503 030424 012737 000000 014526 MOV #0, BOOT ;SPECIFY WHETHER OR NOT TO EXPECT CPU BOOT AND I
2504 030432 004737 032342 JSR PC, MEMSIZ ;FIND ALL THE MEM BOXES
2505 030436 005737 014540 TST EXIT ;WAS ONLY ONE MEM BOX FOUND?
2506 030442 001402 BEQ 1$ ;BRANCH IF NO
2507 030444 000137 031142 JMP TST24 ;WE CAN'T DO THIS TEST
2508 030450 012702 000016 1$: MOV #16, R2 ;POINT TO BOX #7
2509 030454 052777 100000 166346 2$: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2510 030462 012777 000023 163500 MOV #23, @DISPLAY ;SET THE TEST NUMBER
2511 030470 016006 014200 MOV $$STP(R0), SP ;INITIALIZE THE STACK
2512 030474 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
2513 030500 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
2514 030506 005702 TST R2 ;DID WE TEST ALL THE BOXES
2515 030510 002012 BGE 3$ ;BRANCH IF NO
2516 030512 012737 000112 000110 MOV #112, @#110 ;RESTORE LOC 110
2517 030520 005037 000000 CLR @#0 ;RESTORE LOC 0
2518 030524 012737 000001 014540 MOV #1, EXIT ;SIGNAL THE SLAVES TO EXIT

```

```

2519 030532 000137 031142      JMP      TST24      ;GO TO THE NEXT TEST
2520 030536 005762 014502      3$:    TST      START(R2) ;WHAT DO WE KNOW ABOUT THIS BOX?
2521 030542 003037      BGT      10$      ;BRANCH IF NOT THE BASE BOX
2522 030544 001403      BEQ      4$      ;BRANCH IF ITS THE BASE BOX
2523 030546 162702 000002      SUB      #2,      R2 ;THERE WAS NO BOX-POINT TO THE NEXT LOWER BOX
2524 030552 000740      BR       2$      ;CONTINUE
2525 030554 022737 000001 014646      4$:    CMP      #1,      BOXNUM ;WAS THERE ONLY ONE MEM BOX?
2526 030562 002027      BGE      11$      ;BRANCH IF YES
2527 030564 005737 014526      TST      BOOT     ;IS THIS THE DC TEST?
2528 030570 001141      BNE      103$     ;BRANCH IF YES
2529 030572 004737 033042      JSR      PC,      RELOHI ;TO TEST THE BASE BOX,
2530                                ;RELOCATE THE PROGRAM FIRST TO THE NEXT HIGHER B
2531 030576 012737 000001 014532      MOV      #1,      HICORE ;TURN MM ON ON POWER-UP
2532 030604 012737 000001 014534      MOV      #1,      RELOUP ;SIGNAL SLAVES TO RELOCATE
2533 030612 063737 014550 172340      ADD      HIBOX,   KIPARO ;GET READY TO GO TO HIGH CORE
2534 030620 063737 014550 172342      ADD      HIBOX,   KIPAR1
2535 030626 063737 014550 172344      ADD      HIBOX,   KIPAR2
2536 030634 052737 000001 177572      BIS      #1,      MMRO   ;WE ARE NOW IN HIGH CORE
2537 030642                                10$:
2538 030642 005037 014530      11$:    CLR      PATCHK   ;SET TO WRITE PATTERN
2539 030646 004737 032700      JSR      PC,      PATTRN ;WRITE THE PATTERN
2540 030652 004737 032566      JSR      PC,      BUFCLR ;CLEAR THE KEYBOARD BUFFER
2541 030656 104401 042534      TYPE    ,TM106   ;TELL THE OPERATOR TO POWER FAIL THE MEMORY BOX
2542 030662 006202      ASR      R2
2543 030664 010246      MOV      R2,      -(SP)
2544 030666 104405      TYPDS
2545 030670 006302      ASL      R2
2546 030672 104401 042600      TYPE    ,TM107   ;SPECIFY THE CONDITIONS
2547 030676 012703 031040      MOV      #100$,  R3 ;SET UP THE POWER FAIL RETURN
2548 030702 005737 014526      TST      BOOT     ;IS THIS THE DC TEST?
2549 030706 001414      BEQ      12$     ;BRANCH IF NO
2550 030710 012737 020000 000000      13$:    MOV      #20000, @#0 ;MAKE BOOTING SLAVE SP=20000
2551 030716 005237 000110      INC      @#110   ;HANG THE SLAVES BOOT
2552 030722 105777 163262      TSTB    @#TKS    ;IS CHARACTER IN BUFFER?
2553 030726 100370      BPL      13$     ;LOOP
2554 030730 012777 033226 166076      MOV      #ENTR,  @ISTVEC ;SET UP TO INTERRUPT SLAVES
2555 030736 000413      BR       14$     ;CONTINUE
2556 030740                                12$:
2557 030740 105777 163244      TSTB    @#TKS    ;IS CHARACTER IN THE BUFFER?
2558 030744 100375      BPL      12$     ;LOOP
2559 030746 005737 014522      17$:    TST      PWRFL   ;SHOULD THE MASTER HAVE POWER FAILED?
2560 030752 001402      BEQ      18$     ;BRANCH IF NO
2561 030754 104024      ERROR   24      ;FAILURE TO POWER FAIL
2562 030756 000441      BR       102$    ;CONTINUE
2563 030760 104401 047252      18$:    TYPE    ,OK
2564 030764 000436      BR       102$
2565 030766 017704 163166      14$:    MOV      @SWR,   R4 ;GET THE SWITCH VALUES
2566 030772 042704 177760      BIC      #177760, R4 ;SAVE ONLY CPU BITS
2567 030776 012777 000000 166024      MOV      #PGTE,  @ACR  ;ACCESS PGTE REG
2568 031004 010477 166022      MOV      R4,     @ADR  ;SET INTERRUPT BITS
2569 031010 032737 000001 016774      BIT      #BIT0,  CPUACT ;EVEN OR ODD?
2570 031016 001404      BEQ      15$     ;BRANCH IF EVEN
2571 031020 012777 000001 166004      MOV      #1,     @ADR  ;INTERRUPTING AN EVEN NUMBER OF SLAVES
2572 031026 000422      BR       103$
2573 031030 012777 000003 165774      15$:    MOV      #3,     @ADR  ;INTERRUPTING AN ODD # OF SLAVES
2574 031036 000416      BR       103$
  
```

```

2575
2576 031040 005737 014522      100$: TST      PWRFL      ;DID WE EXPECT POWER FAIL?
2577 031044 001002              BNE      101$      ;BRANCH IF YES
2578 031046 104001              ERROR    1         ;MASTER ERROREOUSLY POWER FAILED
2579 031050 000404              BR       102$      ;CONTINUE
2580 031052 005060 017012      101$: CLR      NOPRMP(R0) ;ALLOW CPU IDENTIFICATION
2581 031056 104401 042052      TYPE    ,TM103    ;CPU POWER FAIL MSG
2582 031062 012737 000001 014530 102$: MOV      #1,      PATCHK ;SET UP FOR PATTERN CHECK
2583 031070 004737 032700              JSR     PC,      PATTRN ;CHECK THE PATTERN
2584 031074 032737 000001 177572 103$: BIT      #1,      MMRO   ;ARE WE IN HIGH CORE?
2585 031102 001411              BEQ     104$      ;BRANCH IF NO
2586 031104 004737 033166              JSR     PC,      RELOLO  ;ELSE RELOCATE
2587 031110 012737 000001 014536  MOV     #1,      RELODN ;SIGNAL THE SLAVES
2588 031116 005037 177572              CLR     MMRO      ;WE ARE NOW BACK DOWN IN LOW CORE
2589 031122 005037 014532              CLR     HICORE    ;MAKE SURE MM ON POWER-UP DISABLED
2590 031126 005005              104$: CLR     R5     ;MAKE TS23A DELAY
2591 031130 077501              SOB     R5,      ;
2592 031132 162702 000002              SUB     #2,      R2   ;POINT TO NEXT BOX
2593 031136 000137 030454              JMP     2$        ;CONTINUE
2594
2595
2596
2597
2598
2599

```

```

2600
2601
2602
2603 031142
2604 031142 012777 000024 163020
2605
2606 031150 112761 000024 014002
2607 031156 106237 016776
2608 031162 103375
2609 031164 005237 014546
2610 031170 012737 000001 016776
2611 031176 023737 016774 014546
2612 031204 001374
2613 031206 005037 014540
2614 031212 005037 177776
2615 031216 020027 000000
2616 031222 001001
2617 031224 000474
2618
2619 031226 052777 100000 165574 TS24B:
2620 031234 012777 000024 162726
2621 031242 016006 014200
2622 031246 005060 014612
2623 031252 012760 040630 014702
2624 031260 012703 031372
2625 031264 005737 014540
2626 031270 001404
2627 031272 005037 177572
2628 031276 000137 032164
2629 031302 005737 014534
2630 031306 001421

*****
;*TEST 24 CHECK DC POWERFAIL ON MEM BOXES, CPUS BOOT ON POWER UP
*****
TST24:
MOV #24,@DISPLAY ;SET TEST NUMBER
MOV #24, STSTN(R1) ;SET THE TEST NUMBER
ASRB SYNC.3 ;CONTROL THE ENTRY
BCC 2$
INC ENTR24 ;INCREMENT ENTER FLAG
MOV #1, SYNC.3 ;ALLOW THE OTHERS IN
1$: CMP CPUACT, ENTR24 ;ARE ALL CPUS HERE?
BNE 1$ ;NOT YET
CLR EXIT ;CLEAR THE EXIT FLAG
CLR PSW ;SET KERNAL MODE
CMP R0, #0 ;IS THIS THE MASTER?
BNE TS24B ;BRANCH IF NO
BR TS24A ;THIS IS THE MASTER

TS24B: BIS #BIT15, @ACR ;INITIALIZE THE IIST
MOV #24, @DISPLAY ;SET TEST NUMBER
MOV $$STP(R0), SP ;INITIALIZE THE STACK
CLR PFFT(R0) ;SPECIFY THE POWER FAIL
MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
MOV #100$, R3 ;SET FOR POWER FAIL RETURN
TST EXIT ;FINISHED WITH THIS TEST?
BEQ 1$ ;BRANCH IF NO
CLR MMRO ;MAKE SURE MM IS TURNED OFF
JMP TST25 ;GO TO NEXT TEST
1$: TST RELOUP ;TIME TO RELOCATE?
2$: BEQ 2$ ;BRANCH IF NO

```



```

2631 031310 004737 032604 JSR PC, SETMM ;GET READY FOR RELOCATION
2632 031314 063737 014550 172340 ADD HIBOX, KIPAR0
2633 031322 063737 014550 172342 ADD HIBOX, KIPAR1
2634 031330 063737 014550 172344 ADD HIBOX, KIPAR2
2635 031336 052737 000001 177572 BIS #1, MMRO ;SLAVE IS NOW IN HIGH CORE
2636 031344 005037 014534 CLR RELOUP ;CLEAR RELOCATION FLAG
2637 031350 000726 BR TS24B ;CONTINUE TESTING
2638 031352 005737 014536 2$: TST RELODN ;TIME TO RELOCATE?
2639 031356 001723 BEQ TS24B ;BRANCH IF NO
2640 031360 005037 177572 CLR MMRO ;RETURN TO LOW CORE
2641 031364 005037 014536 CLR RELODN ;CLEAR THE FLAG
2642 031370 000716 BR TS24B ;CONTINUE
2643
2644 031372 005737 014522 100$: TST PWRFL ;SHOULD WE BE HERE?
2645 031376 001002 BNE 101$ ;BRANCH IF YES
2646 031400 104001 ERROR 1 ;UNEXPECTED CPU POWER FAIL
2647 031402 000711 BR TS24B ;CONTINUE TESTING
2648 031404 005060 017012 101$: CLR NOPRMP(R0) ;WANT TO IDENTIFY THE CPU
2649 031410 104401 042052 TYPE ,TM103 ;EXPECTED CPU POWER FAIL
2650 031414 000704 BR TS24B ;CONTINUE TESTING
2651
2652
2653
2654 031416 TS24A: TST R0 ;IS THIS THE MASTER?
2655 031416 005700 BNE 5$ ;BRANCH IF NO
2656 031420 001007 TYPE ,TM77 ;'TEST'
2657 031422 104401 041713 CLR -(SP)
2658 031426 005046 MOV $TSTNM(R1), (SP) ;GET THE TEST NO.
2659 031430 116116 014002 TYPOS .WORD 2 ;TYPE 2 DIGITS, NO LEADING 0
2660 031434 104403
2661 031436 000002
2662 031440
2663 031440 012737 000000 014522 5$: MOV #0, PWRFL ;SPECIFY WHETHER OR NOT TO EXPECT CPU POWER FAIL
2664 031446 012737 000001 014526 MOV #1, BOOT ;SPECIFY WHETHER OR NOT TO EXPECT CPU BOOT AND I
2665 031454 004737 032342 JSR PC, MEMSIZ ;FIND ALL THE MEM BOXES
2666 031460 005737 014540 TST EXIT ;WAS ONLY ONE MEM BOX FOUND?
2667 031464 001402 BEQ 1$ ;BRANCH IF NO
2668 031466 000137 032164 JMP TST25 ;WE CAN'T DO THIS TEST
2669 031472 012702 000016 1$: MOV #16, R2 ;POINT TO BOX #7
2670 031476 052777 100000 165324 2$: BIS #BIT15, @ACR ;INITIALIZE THE IIST
2671 031504 012777 000024 162456 MOV #24, @DISPLAY ;SET THE TEST NUMBER
2672 031512 016006 014200 MOV $$STP(R0), SP ;INIALIZE THE STACK
2673 031516 005060 014612 CLR PFFT(R0) ;SPECIFY THE POWER FAIL
2674 031522 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;SET FOR UNEXPECTED TRAPS TO 4
2675 031530 005702 TST R2 ;DID WE TEST ALL THE BOXES
2676 031532 002012 BGE 3$ ;BRANCH IF NO
2677 031534 012737 000112 000110 MOV #112, @#110 ;RESTORE LOC 110
2678 031542 005037 000000 CLR @#0 ;RESTORE LOC 0
2679 031546 012737 000001 014540 MOV #1, EXIT ;SIGNAL THE SLAVES TO EXIT
2680 031554 000137 032164 JMP TST25 ;GO TO THE NEXT TEST
2681 031560 005762 014502 3$: TST START(R2) ;WHAT DO WE KNOW ABOUT THIS BOX?
2682 031564 003037 BGT 10$ ;BRANCH IF NOT THE BASE BOX
2683 031566 001403 BEQ 4$ ;BRANCH IF ITS THE BASE BOX
2684 031570 162702 000002 SUB #2, R2 ;THERE WAS NO BOX-POINT IO THE NEXT LOWER BOX
2685 031574 000740 BR 2$ ;CONTINUE
2686 031576 022737 000001 014646 4$: CMP #1, BOXNUM ;WAS THERE ONLY ONE MEM BOX?

```


2687	031604	002027				BGE	11\$:BRANCH IF YES
2688	031606	005737	014526			TST	BOOT				:IS THIS THE DC TEST?
2689	031612	001141				BNE	103\$:BRANCH IF YES
2690	031614	004737	033042			JSR	PC,	RELOHI			:TO TEST THE BASE BOX,
2691											:RELOCATE THE PROGRAM FIRST TO THE NEXT HIGHER B
2692	031620	012737	000001	014532		MOV	#1,	HICORE			:TURN MM ON ON POWER-UP
2693	031626	012737	000001	014534		MOV	#1,	RELOUP			:SIGNAL SLAVES TO RELOCATE
2694	031634	063737	014550	172340		ADD	HIBOX,	KIPAR0			:GET READY TO GO TO HIGH CORE
2695	031642	063737	014550	172342		ADD	HIBOX,	KIPAR1			
2696	031650	063737	014550	172344		ADD	HIBOX,	KIPAR2			
2697	031656	052737	000001	177572		BIS	#1,	MMRO			:WE ARE NOW IN HIGH CORE
2698	031664				10\$:						
2699	031664	005037	014530		11\$:	CLR	PATCHK				:SET TO WRITE PATTERN
2700	031670	004737	032700			JSR	PC,	PATTRN			:WRITE THE PATTERN
2701	031674	004737	032566			JSR	PC,	BUFCLR			:CLEAR THE KEYBOARD BUFFER
2702	031700	104401	042534			TYPE	,TM106				:TELL THE OPERATOR TO POWER FAIL THE MEMORY BOX
2703	031704	006202				ASR	R2				
2704	031706	010246				MOV	R2,	-(SP)			
2705	031710	104405				TYPDS					
2706	031712	006302				ASL	R2				
2707	031714	104401	044270			TYPE	,TM110				:SPECIFY THE CONDITIONS
2708	031720	012703	032062			MOV	#100\$,	R3			:SET UP THE POWER FAIL RETURN
2709	031724	005737	014526			TST	BOOT				:IS THIS THE DC TEST?
2710	031730	001414				BEQ	12\$:BRANCH IF NO
2711	031732	012737	020000	000000	13\$:	MOV	#20000,	@#0			:MAKE BOOTING SLAVE SP=20000
2712	031740	005237	000110			INC	@#110				:HANG THE SLAVES BOOT
2713	031744	105777	162240			TSTB	@\$TKS				:IS CHARACTER IN BUFFER?
2714	031750	100370				BPL	13\$:LOOP
2715	031752	012777	033226	165054		MOV	#ENTR,	@ISTVEC			:SET UP TO INTERRUPT SLAVES
2716	031760	000413				BR	14\$:CONTINUE
2717	031762				12\$:						
2718	031762	105777	162222			TSTB	@\$TKS				:IS CHARACTER IN THE BUFFER?
2719	031766	100375				BPL	12\$:LOOP
2720	031770	005737	014522		17\$:	TST	PWRFL				:SHOULD THE MASTER HAVE POWER FAILED?
2721	031774	001402				BEQ	18\$:BRANCH IF NO
2722	031776	104024				ERROR	24				:FAILURE TO POWER FAIL
2723	032000	000441				BR	102\$:CONTINUE
2724	032002	104401	047252		18\$:	TYPE	,OK				
2725	032006	000436				BR	102\$				
2726	032010	017704	162144		14\$:	MOV	@SWR,	R4			:GET THE SWITCH VALUES
2727	032014	042704	177760			BIC	#177760,		R4		:SAVE ONLY CPU BITS
2728	032020	012777	000000	165002		MOV	#PGTE,	@ACR			:ACCESS PGTE REG
2729	032026	010477	165000			MOV	R4,	@ADR			:SET INTERRUPT BITS
2730	032032	032737	000001	016774		BIT	#BIT0,	CPUACT			:EVEN OR ODD?
2731	032040	001404				BEQ	15\$:BRANCH IF EVEN
2732	032042	012777	000001	164762		MOV	#1,	@ADR			:INTERRUPTING AN EVEN NUMBER OF SLAVES
2733	032050	000422				BR	103\$:
2734	032052	012777	000003	164752	15\$:	MOV	#3,	@ADR			:INTERRUPTING AN ODD # OF SLAVES
2735	032060	000416				BR	103\$:
2736											
2737	032062	005737	014522		100\$:	TST	PWRFL				:DID WE EXPECT POWER FAIL?
2738	032066	001002				BNE	101\$:BRANCH IF YES
2739	032070	104001				ERROR	1				:MASTER ERRONEOUSLY POWER FAILED
2740	032072	000404				BR	102\$:CONTINUE
2741	032074	005060	017012		101\$:	CLR	NOPRMP(R0)				:ALLOW CPU IDENTIFICATION
2742	032100	104401	042052			TYPE	,TM103				:CPU POWER FAIL MSG

```

2743 032104 012737 000001 014530 102$: MOV #1, PATCHK ;SET UP FOR PATTERN CHECK
2744 032112 004737 032700 JSR PC, PATTRN ;CHECK THE PATTERN
2745 032116 032737 000001 177572 103$: BIT #1, MMRO ;ARE WE IN HIGH CORE?
2746 032124 001411 BEQ 104$ ;BRANCH IF NO
2747 032126 004737 033166 JSR PC, RELOLO ;ELSE RELOCATE
2748 032132 012737 000001 014536 MOV #1, RELODN ;SIGNAL THE SLAVES
2749 032140 005037 177572 CLR MMRO ;WE ARE NOW BACK DOWN IN LOW CORE
2750 032144 005037 014532 CLR HICORE ;MAKE SURE MM ON POWER-UP DISABLED
2751 032150 005005 104$: CLR R5 ;MAKE TS24A DELAY
2752 032152 077501 SOB R5, .
2753 032154 162702 000002 SUB #2, R2 ;POINT TO NEXT BOX
2754 032160 000137 031476 JMP 2$ ;CONTINUE
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764

```

```

:*****
:*TEST 25 CHECK SYSTEM RECOVERY ON AC POWER FAIL
:*****

```

```

2765 032164 TST25:
2766 032164 012777 000025 161776 MOV #25,@DISPLAY ;SET TEST NUMBER
2767 032172 112761 000025 014002 MOVB #25, $STNM(R1) ;SET THE TEST NUMBER
2768 032200 032770 000400 014160 BIT #SW08, @SWR(R0) ;SKIP THIS TEST?
2769 032206 001002 BNE 1$ ;BRANCH IF NO
2770 032210 000137 035326 JMP $EOP ;ELSE GO TO EOP
2771 032214 1$:
2772 032214 020027 000000 CMP R0,#0 ;SWITCH THIS PROCESSOR?
2773 032220 001002 BNE 64$ ;BRANCH IF NO
2774 032222 000137 032232 JMP TS25A ;YES, SWITCH TO A.
2775 032226 000137 032264 64$: JMP TS25B ;NO, SWITCH TO B.
2776 032232 TS25A:
2777 032232 005700 TST R0 ;IS THIS THE MASTER?
2778 032234 001007 BNE 5$ ;BRANCH IF NO
2779 032236 104401 041713 TYPE ,TM77 ;'TEST'
2780 032242 005046 CLR -(SP)
2781 032244 116116 014002 MOVB $STNM(R1),(SP) ;GET THE TEST NO.
2782 032250 104403 TYPOS
2783 032252 000002 .WORD 2 ;TYPE 2 DIGITS, NO LEADING 0
2784 032254 5$:
2785 032254 104401 014333 TYPE ,$CRLF
2786 032260 104401 042115 TYPE ,TM104 ;'POWER FAIL ENTIRE SYSTEM'
2787 032264 012760 004000 014612 TS25B: MOV #TI, PFFT(R0)
2788 032272 005003 CLR R3
2789 032274 005037 014714 CLR SYNC.2
2790 032300 000001 WAIT ;WAIT FOR POWER TO FAIL
2791 032302 106237 016776 3$: ASRB SYNC.3 ;CONTROL THE CPUS
2792 032306 103375 BCC 3$
2793 032310 005237 014714 INC SYNC.2 ;COUNT THE CPUS
2794 032314 012737 000001 016776 MOV #1, SYNC.3 ;ALLOW ANOTHER IN
2795 032322 023737 016774 014714 1$: CMP CPUACT, SYNC.2
2796 032330 001374 BNE 1$
2797 032332 104401 014220 TYPE ,NULL
2798 032336 000137 035326 JMP $EOP ;GO TO END OF PASS

```

```

2799
2800
2801 032342 005002 MEMSIZ: .SBTTL MEMORY BOX TEST ROUTINES
2802 032344 012704 172100 CLR R2 ;GET SET TO FILL THE START AND STOP TABLES
2803 032350 005037 014646 MOV #172100, R4 ;POINT TO THE FIRST CSR
2804 032354 CLR BOXNUM ;START WITH 0 BOXES
2805 032354 012760 032476 014702 1$: MOV #100$, ERRTAB(R0) ;SET UP FOR NO BOX
2806 032362 052714 000010 BIS #10, (R4) ;SET UP TO GET BOX CAPACITY
2807 032366 012462 014462 MOV (R4)+, STOP(R2)
2808 032372 000362 014462 SWAB STOP(R2)
2809 032376 042762 177600 014462 BIC #177600, STOP(R2);NOW WE HAVE IT
2810 032404 012462 014502 MOV (R4)+, START(R2) ;NOW GET THE STARTING ADR.
2811 032410 042762 177000 014502 BIC #177000, START(R2)
2812 032416 016205 014502 MOV START(R2), R5 ;MAKE IT LOOK LIKE A PAR
2813 032422 072527 000012 ASH #10., R5
2814 032426 010562 014502 MOV R5, START(R2)
2815 032432 016205 014462 MOV STOP(R2), R5 ;DO THE SAME FOR STOP
2816 032436 072527 000012 ASH #10., R5
2817 032442 010562 014462 MOV R5, STOP(R2)
2818 032446 066262 014502 014462 ADD START(R2), STOP(R2);STOP=START+CAPACITY
2819 032454 005237 014646 INC BOXNUM ;INCREMENT BOX COUNT
2820 032460 005762 014502 TST START(R2) ;IS THIS THE BASE BOX?
2821 032464 001413 BEQ 101$ ;BRANCH IF YES
2822 032466 016237 014502 014550 MOV START(R2), HIBOX ;GET A BOX TO RELOCATE TO
2823 032474 000407 BR 101$
2824 032476 012762 177777 014502 100$: MOV #-1, START(R2) ;INDICATE NON-EXISTENT BOX
2825 032504 062706 000004 ADD #4, SP ;RESTORE THE STACK
2826 032510 062704 000004 ADD #4, R4 ;POINT TO NEXT CSR PAIR
2827 032514 062702 000002 101$: ADD #2, R2 ;POINT TO NEXT TABLE LOCATIONS
2828 032520 022702 000020 CMP #20, R2 ;HAVE WE LOOKED FOR EIGHT BOXES?
2829 032524 003313 BGT 1$ ;BRANCH IF NO
2830 032526 012760 040630 014702 MOV #CPUER, ERRTAB(R0) ;RESET TRAP TO 4 POINTER
2831 032534 022737 000001 014646 CMP #1, BOXNUM ;WAS THERE MORE THAN A SINGLE BOX?
2832 032542 002410 BLT 102$ ;BRANCH IF YES
2833 032544 005737 014522 TST PWRFL ;IS THIS TEST 23?
2834 032550 001005 BNE 102$ ;BRANCH IF YES
2835 032552 104401 044206 TYPE ,TM109 ;PRINT ONLY ONE BOX MSG
2836 032556 012737 000001 014540 MOV #1, EXIT ;SIGNAL TO EXIT
2837
2838 032564 000207 102$: RTS PC ;RETURN
2839
2840 032566 105777 161416 BUFCLR: TSTB @STKS ;IS THE BUFFER EMPTY?
2841 032572 100003 BPL 1$ ;BRANCH IF YES
2842 032574 117705 161412 MOVB @STKB, R5 ;FLUSH IT
2843 032600 000772 BR BUFCLR ;LOOP
2844 032602 000207 1$: RTS PC ;RETURN
2845
2846 032604 005037 172340 SETMM: CLR KIPAR0 ;SET UP PARS 0,1,2,7
2847 032610 012737 077406 172300 MOV #77406, KIPDR0
2848 032616 012737 000200 172342 MOV #200, KIPAR1
2849 032624 012737 077406 172302 MOV #77406, KIPDR1
2850 032632 012737 000400 172344 MOV #400, KIPAR2
2851 032640 012737 077406 172304 MOV #77406, KIPDR2
2852 032646 012737 177600 172356 MOV #177600, KIPAR7
2853 032654 012737 077406 172316 MOV #77406, KIPDR7
2854 032662 012737 000020 172516 MOV #20, MMR3
    
```

```

2855 032670 012737 033534 000250      MOV      #MMERR, MMVEC
2856 032676 000207      RTS      PC
2857
2858 032700 005737 014526      PATRN:  TST      BOOT      ;ARE WE DOING THE DC TEST?
2859 032704 001055          BNE      7$      ;BRANCH IF YES
2860 032706 005762 014502      TST      START(R2) ;ARE WE DOING THE BASE BOX?
2861 032712 001452          BEQ      7$      ;BRANCH IF YES
2862 032714 004737 032604      JSR      PC,      SETMM   ;SET UP PARS 0,1,2,7
2863 032720 016237 014502 172346      MOV      START(R2), KIPAR3 ;USE PAR3 TO WRITE PATRN
2864 032726 012737 077406 172306      MOV      #77406, KIPDR3
2865 032734 012760 040630 014702      MOV      #CPUER, ERRTAB(R0) ;SET UP FOR UNEXPECTED TRAPS
2866 032742 052737 000001 177572      BIS      #1,      MMRO   ;TURN ON MM
2867 032750 012705 060000          1$:      MOV      #60000, R5     ;POINT TO PAR3 SPACE
2868 032754 026237 014462 172346      CMP      STOP(R2), KIPAR3 ;IS THIS THE END OF THE BOX?
2869 032762 003424          BLE      6$      ;BRANCH IF YES
2870 032764 005737 014530          2$:      TST      PATCHK     ;ARE WE WRITING A PATTERN?
2871 032770 001002          BNE      3$      ;BRANCH IF ONLY READING IT
2872 032772 012715 152525          MOV      #152525, (R5)   ;WRITE THE PATTERN
2873 032776 022725 152525          3$:      CMP      #152525, (R5)+ ;IS THE PATTERN CORRECT?
2874 033002 001405          BEQ      4$      ;BRANCH IF YES
2875 033004 013760 172346 014250      MOV      KIPAR3, $REG1(R0) ;SAVE THE BAD ADDRESS (PAR)
2876 033012 104013          ERROR   13      ;MEMORY IS CORRUPTED
2877 033014 000407          BR      6$      ;REPORT ONLY ONE ERROR
2878 033016 022705 100000          4$:      CMP      #100000, R5   ;ARE WE STILL IN PAR3 SPACE?
2879 033022 003360          BGT      2$      ;BRANCH IF YES
2880 033024 062737 000200 172346      ADD      #200, KIPAR3   ;ELSE RESET THE PAR
2881 033032 000746          BR      1$      ;AND CONTINUE
2882 033034
2883 033034 005037 177572          6$:      CLR      MMRO      ;END MM
2884 033040 000207          7$:      RTS      PC      ;RETURN
2885
2886 033042 004737 032604          RELOHI: JSR      PC,      SETMM   ;SET UP PARS 0,1,2,7
2887 033046 012760 040630 014702      MOV      #CPUER, ERRTAB(R0) ;SET UP FOR TRAP TO 4
2888 033054 013737 014550 172346      MOV      HIBOX, KIPAR3   ;PARS 3,4,5 WILL TAKE US TO HIGH CORE
2889 033062 012737 077406 172306      MOV      #77406, KIPDR3
2890 033070 013737 014550 172350      MOV      HIBOX, KIPAR4
2891 033076 062737 000200 172350      ADD      #200, KIPAR4
2892 033104 012737 077406 172310      MOV      #77406, KIPDR4
2893 033112 013737 014550 172352      MOV      HIBOX, KIPAR5
2894 033120 062737 000400 172352      ADD      #400, KIPAR5
2895 033126 012737 077406 172312      MOV      #77406, KIPDR5
2896 033134 012704 060000          MOV      #60000, R4     ;POINT TO PAR3
2897 033140 005005          CLR      R5         ;POINT TO PAR0
2898 033142 052737 000001 177572      BIS      #1,      MMRO   ;TURN ON MM
2899 033150 012524          1$:      MOV      (R5)+, (R4)+   ;RELOCATE THE PROGRAM
2900 033152 022705 060000          CMP      #60000, R5     ;ARE WE FINISHED?
2901 033156 003374          BGT      1$      ;BRANCH IF NO
2902 033160 005037 177572          CLR      MMRO      ;TURN OFF MM
2903 033164 000207          RTS      PC
2904
2905 033166 005037 172346          RELOLO: CLR      KIPAR3   ;USE PARS 3,4,5 TO RESTORE CODE TO LOW CORE
2906 033172 012737 000200 172350      MOV      #200, KIPAR4
2907 033200 012737 000400 172352      MOV      #400, KIPAR5
2908 033206 012704 060000          MOV      #60000, R4     ;POINT TO PAR 3
2909 033212 005005          CLR      R5         ;POINT TO PAR 0
2910 033214 012524          1$:      MOV      (R5)+, (R4)+   ;RESTORE THE PROGRAM

```

```

2911 033216 022705 060000      CMP      #60000, R5      ;ARE WE FINISHED?
2912 033222 003374      BGT      1$           ;BRANCH IF NO
2913 033224 000207      RTS      PC           ;RETURN-STILL EXECUTING IN HIGH CORE
2914
2915 033226 052737 001000 177746 ENTR:  BIS      #BIT9, CONTRL ;TURN OFF CACHE
2916 033234 017705 163570      MOV      @ACR, R5     ;GET CPU ID
2917 033240 072527 177770      ASH      #-10, R5
2918 033244 005004      CLR      R4           ;SET UP R0
2919 033246 026405 014226 65$:  CMP      $CPUID(R4), R5
2920 033252 001404      BEQ      64$
2921 033254 005724      TST      (R4)+
2922 033256 020427 000010      CMP      R4, #10
2923 033262 002771      BLT      65$
2924 033264 010400 64$:  MOV      R4, R0
2925 033266 010001      MOV      R0, R1      ;SET UP R1
2926 033270 006201      ASR      R1
2927 033272 052777 100000 163530  BIS      #BIT15, @ACR ;INITIALIZE THE IIST
2928 033300 016006 014200      MOV      $$STP(R0), SP ;SET UP THE STACK
2929 033304 012760 034656 014662  MOV      #SPWRDN, PWRTAB(R0);SET UP FOR POWER DOWN
2930 033312 020027 000000      CMP      R0, #0
2931 033316 001406      BEQ      1$           ;SHOULD WE BE HERE?
2932 033320 005060 017012      CLR      NOPRMP(R0) ;BRANCH IF NO
2933 033324 104401 045163      TYPE      ,TM11
2934 033330 000137 031226      JMP      TS24B
2935 033334 104025 1$:  ERROR  25           ;RETURN SLAVES
2936 033336 000000      HALT                ;UNEXPECTED INTERRUPT
2937
2938
2939
2940
2941      .SBTTL  PARITY ERROR HANDLER
2942
2943 033340 005327  PARERR: DEC      (PC)+ ;FIRST TIME IN?
2944 033342 000001  PARFLG: .WORD  1
2945 033344 002001      BGE      1$           ;BRANCH IS YES
2946 033346 000000      HALT                ;ELSE HALT
2947 033350 013760 177766 014250 1$:  MOV      CPUERR, $REG1(R0)
2948 033356 013760 177744 014260      MOV      MEMERR, $REG2(R0)
2949 033364 104007      ERROR  7           ;UNEXPECTED TRAP TO 114
2950 033366 000000      HALT
2951 033370 013737 177744 177744      MOV      @MEMERR, @MEMERR ;CLEAR ERROR INDICATORS
2952 033376 012737 000001 033342      MOV      #1, PARFLG ;INITIALIZE PARITY ERROR FLAG
2953 033404 000002      RTI                ;CONTINUE TESTING
2954
2955      .SBTTL  SETUP MEMORY MANAGMENT REGISTERS
2956 033406 012737 000000 172340 MAP:  MOV      #0, @KIPAR0 ;SETUP PAR0 FOR 1ST 4K
2957 033414 012737 077406 172300      MOV      #77406, @KIPDR0 ;4K, R/W, EXPAND UP
2958 033422 012737 000200 172342      MOV      #200, @KIPAR1 ;SETUP PAR0 FOR 2ND 4K
2959 033430 012737 077406 172302      MOV      #77406, @KIPDR1 ;4K, R/W, EXPAND UP
2960 033436 012737 000400 172344      MOV      #400, @KIPAR2 ;SETUP PAR2 FOR NEXT 4K
2961 033444 012737 077406 172304      MOV      #77406, @KIPDR2 ;4K, R/W, EXPAND UP
2962 033452 012737 000000 172352      MOV      #0, @KIPAR5 ;SET UP PAR5 FOR 1ST 4K
2963 033460 012737 077406 172312      MOV      #77406, @KIPDR5 ;4K, R/W, ED=UP
2964 033466 012737 000200 172354      MOV      #200, @KIPAR6 ;SET UP PAR6 FOR 2ND 4K
2965 033474 012737 000000 172314      MOV      #0, @KIPDR6 ;ABORT ALL REFERENCES
2966 033502 012737 177600 172356      MOV      #177600, @KIPAR7 ;SET UP PAR7 FOR I/O PAGE
    
```

```

2967 033510 012737 077406 172316      MOV      #77406,@#KIPDR7 ;4K, R/W, ED=UP
2968 033516 012737 000020 172516      MOV      #BIT04,@#MMR3  ;SET UP FOR 22-BIT MAPPING
2969 033524 012737 033534 000250      MOV      #MMERR,@#MMVEC ;SET UP MEMORY MANAGEMENT VECTOR
2970 033532 000207                RTS      PC              ;RETURN FROM CALL
2971 033534 000000      MMERR: HALT            ;MEMORY MANAGEMENT ERROR
2972 033536 000776      BR      MMERR
2973
2974
2975 033540                .SBTTL MASSBUS TRANSFER ROUTINES
2976 033540 032737 000200 176700      4$: BIT      #BIT7, @#RPCS1 ;WAIT FOR CONTROLLER READY
2977 033546 001774                BEQ     4$
2978 033550 156037 014632 176710      BISB    MBDSW(R0), @#RPCS2 ;GET THE DRIVE #
2979 033556 012737 174000 176702      MOV      #-4000,@#RPWC  ;:SET WORD COUNT
2980 033564 010437 176704                MOV      R4,@#RPBA      ;:SET MEMORY ADDRESS
2981 033570 005037 176706                CLR     @#RPDA          ;:READ SECTOR 0
2982 033574 032737 000200 176712      3$: BIT      #BIT7, @#RPDS ;WAIT FOR DRIVE READY
2983 033602 001774                BEQ     3$
2984 033604 052737 000001 176700      BIS     #BIT0,@#RPCS1  ;:
2985 033612 106237 033534                1$: ASRB    MMERR        ;:DO ASRB DURING TRANSFER
2986 033616 106237 033534                ASRB    MMERR
2987 033622 106237 033534                ASRB    MMERR
2988 033626 106237 033534                ASRB    MMERR
2989 033632 106237 033534                ASRB    MMERR
2990 033636 106237 033534                ASRB    MMERR
2991 033642 106237 033534                ASRB    MMERR
2992 033646 106237 033534                ASRB    MMERR
2993 033652 106237 033534                ASRB    MMERR
2994 033656 106237 033534                ASRB    MMERR
2995 033662 032737 000200 176712      BIT     #BIT7,@#RPDS  ;:DEVICE READY?
2996 033670 001750                BEQ     1$             ;:BRANCH IF NO.
2997 033672 005737 176700      TST     @#RPCS1      ;:ANY ERRORS?
2998 033676 100001                BPL     2$             ;:NO
2999 033700 000000                HALT    ;:YES
3000 033702 000207                2$: RTS      PC        ;:RETURN
3001
3002 033704 012737 033724 000100      SETCLK: .SBTTL LINE CLOCK ROUTINE
3003 033712 005204                MOV     #5$,@#100    ;:SET THE INTERRUPT VECTOR FOR CLK
3004 033714 012737 000100 177546      INC     R4           ;:ADD 1 TO THE ARGUMENT PASSED
3005 033722 000207                MOV     #BIT6,@#LKS  ;:START THE CLOCK
3006 033724 052737 000340 177776      5$: RTS      PC        ;:RETURN
3007 033732 042737 000200 177546      BIS     #340,@#PS    ;:HIGH PRIORITY
3008 033740 005304                BIC     #BIT7,@#LKS  ;:CLEAR THE MONITOR BIT
3009 033742 005704                DEC     R4           ;:ONE TICK
3010 033744 001010                TST     R4           ;:COUNT TO ZERO?
3011 033746 005037 177546      BNE     6$           ;:NO DON'T STOP THE CLOCK
3012 033752 000240                CLR     @#LKS        ;:TURN IT OFF
3013 033754 000240                NOP
3014 033756 000240                NOP
3015 033760 000240                NOP
3016 033762 062716 000004                ADD     #4,(SP)      ;:SKIP RETURN
3017 033766 162716 000002      6$: SUB     #2,(SP)   ;:IF COUNT ISN'T EXPIRED...
3018 033772 000002                7$: RTI
3019
3020
3021
3022

```

```

3023      .SBTTL  POWER FAIL ROUTINE (SECTION 1)
3024
3025 033774 012737 034436 014662 POWDWN: MOV      #ILLUP,PWRTAB ;IF TOO FAST WITH POWER UP
3026 034002 105737 016740          TSTB     UBEF      ;;USE UNIBUS EXERCISER?
3027 034006 001403          BEQ      64$
3028 034010 042737 000020 170016          BIC      #BIT4,@#UBCR2 ;;CLEAR POWER FAIL ENABLE
3029 034016          64$:
3030 034016 022706 000440          CMP      #440,SP      ;YELLOW OR RED?
3031 034022 100402          BMI      1$          ;NO
3032 034024 016006 014200          MOV      $$STP(R0),SP ;SET EMERGENCY STACK
3033 034030 010246          1$:  MOV      R2,-(SP)      ;SAVE R2
3034 034032 012702 177572          MOV      #MMR0,R2      ;SAVE PSW THRU MMR0
3035 034036 012246          10$: MOV      (R2)+,-(SP)
3036 034040 022702 177676          CMP      #UDPAR7,R2
3037 034044 103374          BHIS     10$
3038 034046 013746 177776          MOV      @#PSW,-(SP)   ;SAVE PSW
3039 034052 013746 177746          MOV      @#CONTRL,-(SP) ;SAVE CACHE CONTROL
3040 034056 013746 172516          MOV      @#MMR3,-(SP)  ;SAVE MMR3
3041 034062 012702 172200          MOV      #SIPDR0,R2    ;SAVE SIPDR0 THRU KDPAR7
3042 034066 012246          20$: MOV      (R2)+,-(SP)
3043 034070 022702 172376          CMP      #KDPAR7,R2
3044 034074 103374          BHIS     20$
3045 034076 012702 170200          MOV      #MAPL00,R2    ;SAVE THE UNIBUS MAP
3046 034102 012246          30$: MOV      (R2)+,-(SP)
3047 034104 022702 170376          CMP      #MAPH37,R2
3048 034110 103374          BHIS     30$
3049 034112 010046          MOV      R0,-(SP)     ;SAVE THE GENERAL REGISTERS
3050 034114 010146          MOV      R1,-(SP)
3051 034116 010346          MOV      R3,-(SP)
3052 034120 010446          MOV      R4,-(SP)
3053 034122 010546          MOV      R5,-(SP)
3054 034124 010637 014572          MOV      SP,SAV6      ;SAVE THE STACK
3055 034130 005060 016754          CLR      COUNT0(R0)   ;CLEAR LOOP COUNTER
3056 034134 060000          ADD      R0,R0        ;INDEX TO THE RIGHT COUNTER
3057 034136 060000          ADD      R0,R0
3058 034140 060000          ADD      R0,R0
3059 034142 160100          SUB      R1,R0
3060 034144 160100          SUB      R1,R0
3061 034146 012737 034256 014662          MOV      #POWUP,PWRTAB ;ENABLE GOOD POWER-UP
3062 034154 012737 000001 016752          MOV      #1,LOOPS     ;# OF LOOPS FOR 2 MS (WORST CASE CONTENTION)
3063 034162 000160 034166          JMP      3$(R0)        ;START LOOPING UNTIL DC POWER FAILS
3064 034166 005237 016754          3$:  INC      COUNT0
3065
3066 034172 023737 016752 016754          CMP      LOOPS,COUNT0 ;2MS UP?
3067 034200 001372          BNE      3$           ;BRANCH IF NO
3068 034202 000000          HALT     ;FINISHED
3069 034204 005237 016756          4$:  INC      COUNT1
3070 034210 023737 016757 016756          CMP      LOOPS,COUNT1
3071 034216 001372          BNE      4$
3072 034220 000000          HALT
3073 034222 005237 016760          5$:  INC      COUNT2
3074 034226 023737 016752 016760          CMP      LOOPS,COUNT2
3075 034234 001372          BNE      5$
3076 034236 000000          HALT
3077 034240 005237 016762          6$:  INC      COUNT3
3078 034244 023737 016752 016762          CMP      LOOPS,COUNT3

```



```

3079 034252 001372          BNE      6$
3080 034254 000000          HALT
3081
3082 034256                POWUP:
3083 034256 012737 034442 014662  MOV     #ILLDWN,PWRTAB;SET TOO FAST DOWN VECTOR
3084 034264 013706 014572  MOV     SAV6,SP          ;RESET SP
3085 034270 012605          MOV     (SP)+,R5          ;RESTORE THE REC STERS
3086 034272 012604          MOV     (SP)+,R4
3087 034274 012603          MOV     (SP)+,R3
3088 034276 012601          MOV     (SP)+,R1
3089 034300 012600          MOV     (SP)+,R0
3090 034302 012702 170400  MOV     #MAPH37+2,R2     ;RESTORE UNIB' AP
3091 034306 012642 10$:   MOV     (SP)+,-(R2)
3092 034310 022702 170200  CMP     #MAPL00,R2
3093 034314 103774          BLO    10$
3094 034316 012702 172400  MOV     #KDPAR7+2,R2     ;RESTORE K AND S PARS/PDRS
3095 034322 012642 20$:   MOV     (SP)+,-(R2)
3096 034324 022702 172200  CMP     #SIPDR0,R2
3097 034330 103774          BLO    20$
3098 034332 012637 172516  MOV     (SP)+,@MMR3      ;RESTORE MMR3
3099 034336 012637 177746  MOV     (SP)+,@CONTRL    ;RESTORE CACHE CONTRL
3100 034342 012637 177776  MOV     (SP)+,@PSW       ;RESTORE PSW
3101 034346 012702 177700  MOV     #UDPAR7+2,R2     ;RESTORE PSW THRU MMRO
3102 034352 012642 30$:   MOV     (SP)+,-(R2)
3103 034354 022702 177572  CMP     #MMR0,R2
3104 034360 103774          BLO    30$
3105 034362 012602          MOV     (SP)+,R2         ;RESTORE R2
3106 034364 004737 034446  JSR     PC,TIMIT         ;CHECK THE POWER-DOWN TIME
3107 034370 012737 033774 014662  MOV     #POWDWN,PWRTAB   ;RESET THE DOWN VECTOR
3108 034376 105737 016740  TSTB   UBEF              ;UBE BEING USED?
3109 034402 001403          BEQ    2$                ;BRANCH IF NO
3110 034404 012737 000001 017022  MOV     #1,UBELCK        ;CLEAR THE PF LOCK
3111 034412
3112 034412 105737 016737 2$:   TSTB   MPF                ;MULTIPROCESSOR MODE?
3113 034416 001403          BEQ    1$                ;BRANCH IF NO
3114 034420 052737 001000 177746  BIS     #1000,@CONTRL    ;BYPASS CACHE
3115 034426 052737 000014 177746 1$:   BIS     #14,@CONTRL      ;TURN OFF CACHE
3116 034434 000113          JMP     (R3)              ;JUMP INDIRECT TO R3
3117
3118 034436 000000  ILLUP: HALT              ;POWER UP BEFORE POWER DOWN COMPLETE
3119 034440 000776          BR     .-2                ;LOCK UP THE HALT
3120
3121 034442 000000  ILLDWN: HALT             ;POWERED DOWN BEFORE UP COMPLETE
3122 034444 000776          BR     .-2                ;LOCK UP THE HALT
3123
3124
3125 034446                TIMIT:
3126 034446 023760 016752 016754  CMP     LOOPS,COUNT0(R0) ;DID WE HAVE ENOUGH POWER DOWN TIME?
3127 034454 001402          BEQ    1$                ;BRANCH IF YES
3128 034456 104401 045221          TYPE   ,SDOWN            ;NOT ENOUGH TIME
3129
3130 034462 000207 1$:   RTS     PC
3131
3132
3133
3134

```



```

3135          .SBTTL POWER FAIL ROUTINE (SECTION 2)
3136 034464          $POWER:
3137 034464 005737 014532      TST      HICORE          ;DID WE POWER DOWN IN HIGH CORE?
3138 034470 001451          BEQ      1$              ;BRANCH IF NO
3139 034472 005037 172340      CLR      KIPARO          ;SET UP PARS 0,1,2,7
3140 034476 012737 077406 172300  MOV     #77406, KIPDR0
3141 034504 012737 000200 172342  MOV     #200, KIPAR1
3142 034512 012737 077406 172302  MOV     #77406, KIPDR1
3143 034520 012737 000400 172344  MOV     #400, KIPAR2
3144 034526 012737 077406 172304  MOV     #77406, KIPDR2
3145 034534 012737 177600 172356  MOV     #177600, KIPAR7
3146 034542 012737 077406 172316  MOV     #77406, KIPDR7
3147 034550 012737 000020 172516  MOV     #20, MMR3
3148 034556 012737 033534 000250  MOV     #MMERR, MMVEC
3149 034564 063737 014550 172340  ADD     HIBOX, KIPAR0
3150 034572 063737 014550 172342  ADD     HIBOX, KIPAR1
3151 034600 063737 014550 172344  ADD     HIBOX, KIPAR2
3152 034606 052737 000001 177572  BIS     #1, MMRO          ;POWER-UP IN HIGH CORE
3153 034614          1$:
3154 034614 017705 162210      MOV     @ACR, R5          ;COPY ACR
3155 034620 072527 177770      ASH     #-10, R5        ;GET THE ID
3156 034624 005004          CLR     R4              ;SET UP R0
3157 034626 026405 014226      65$:  CMP     $CPUID(R4), R5
3158 034632 001404          BEQ     64$
3159 034634 005724          TST     (R4)+
3160 034636 020427 000010      CMP     R4, #10
3161 034642 002771          BLT     65$
3162 034644 010400      64$:  MOV     R4, R0
3163 034646 010001          MOV     R0, R1
3164 034650 006201          ASR     R1
3165 034652 000170 014662      JMP     @PWRTAB(R0)      ;JUMP TO THE POWER ROUTINE
3166
3167
3168 034656 012760 035316 014662  $PWRDN: MOV    #SILLUP,PWRTAB(R0) ;SET UVECT FOR ILLEGAL UP
3169 034664 032760 000040 014612  BIT     #SSD,PFFT(R0)   ;SEND A SIGNAL?
3170 034672 001403          BEQ     10$            ;NO
3171 034674 012737 000001 014642  MOV     #1, SIGNAL
3172
3173          10$:
3174 034702 026006 017052      CMP     YELLIM(R0),SP   ;YELLOW OR RED?
3175 034706 100402          BMI     1$            ;NO
3176 034710 016006 014200      MOV     $$SSTP(R0),SP  ;SET EMERGENCY STACK
3177 034714 010246      1$:  MOV     R2,-(SP)        ;SAVE R2
3178 034716 012702 177572      MOV     #MMRO,R2       ;SAVE PSW THRU MMRO
3179 034722 012246      100$: MOV     (R2)+,-(SP)
3180 034724 022702 177676      CMP     #UDPAR7,R2
3181 034730 103374          BHIS   100$
3182 034732 013746 177776      MOV     @PSW,-(SP)     ;SAVE PSW
3183 034736 013746 177746      MOV     @CONTRL,-(SP)  ;SAVE CACHE CONTRL
3184 034742 013746 172516      MOV     @MMR3,-(SP)   ;SAVE MMR3
3185 034746 012702 172200      MOV     #SIPDR0,R2    ;SAVE SIPDR0 THRU KDPAR7
3186 034752 012246      20$:  MOV     (R2)+,-(SP)
3187 034754 022702 172376      CMP     #KDPAR7,R2
3188 034760 103374          BHIS   20$
3189 034762 012702 170200      MOV     #MAPL00,R2    ;SAVE THE UNIBUS MAP
3190 034766 012246      30$:  MOV     (R2)+,-(SP)

```

```

3191 034770 022702 170376      CMP      #MAPH37,R2
3192 034774 103374      BHS     30$
3193 034776 010046      MOV     R0,-(SP)      ;SAVE THE GENERAL REGISTERS
3194 035000 010146      MOV     R1,-(SP)
3195 035002 010346      MOV     R3,-(SP)
3196
3197 035004 010446      MOV     R4,-(SP)
3198 035006 010546      MOV     R5,-(SP)
3199 035010 010660 014572      MOV     SP, SAV6(R0)      ;SAVE THE STACK
3200 035014 005060 016754 40$:      CLR     COUNT0(R0)      ;CLEAR THE LOOP COUNTER
3201 035020 012760 035142 014662      MOV     #SPWRUP, PWRTAB(R0);GET SET FOR POWER-UP
3202 035026 060000      ADD     R0, R0      ; INDEX TO THE RIGHT COUNTER
3203 035030 060000      ADD     R0, R0
3204 035032 060000      ADD     R0,R0
3205 035034 160100      SUB     R1,R0
3206 035036 160100      SUB     R1,R0
3207 035040 012737 000001 016752      MOV     #1,LOOPS      ;# OF LOOPS FOR 2 MS (WORST CASE CONTENTION)
3208 035046 000160 035052      JMP     3$(R0)      ;START LOOPING UNTIL DC POWER FAILS
3209 035052 005237 016754 3$:      INC     COUNT0
3210
3211 035056 023737 016752 016754      CMP     LOOPS,COUNT0      ;2MS UP?
3212 035064 001372      BNE     3$      ;BRANCH IF NO
3213 035066 000000      HALT    ;FINISHED
3214 035070 005237 016756 4$:      INC     COUNT1
3215 035074 023737 016752 016756      CMP     LOOPS,COUNT1
3216 035102 001372      BNE     4$
3217 035104 000000      HALT
3218 035106 005237 016760 5$:      INC     COUNT2
3219 035112 023737 016752 016760      CMP     LOOPS,COUNT2
3220 035120 001372      BNE     5$
3221 035122 000000      HALT
3222 035124 005237 016762 6$:      INC     COUNT3
3223 035130 023737 016752 016762      CMP     LOOPS,COUNT3
3224 035136 001372      BNE     6$
3225 035140 000000      HALT
3226
3227 035142 012760 035322 014662 $PWRUP: MOV     #ILLDN,PWRTAB(R0)      ;SET VECTOR FOR FAST DOWN
3228 035150 016006 014572      MOV     SAV6(R0),SP      ;RESTORE STACK
3229 035154 012605      MOV     (SP)+,R5      ;SAVE THE GENERAL REGISTERS
3230 035156 012604      MOV     (SP)+,R4
3231 035160 012603      MOV     (SP)+,R3
3232 035162 012601      MOV     (SP)+,R1
3233 035164 012600      MOV     (SP)+,R0
3234 035166 012702 170400      MOV     #MAPH37+2,R2      ;RESTORE UNIBUS MAP
3235 035172 012642 10$:      MOV     (SP)+,-(R2)
3236 035174 022702 170200      CMP     #MAPL00,R2
3237 035200 103774      BLO     10$
3238 035202 012702 172400      MOV     #KDPAR7+2,R2      ;RESTORE K AND S PARS/PDRS
3239 035206 012642 20$:      MOV     (SP)+,-(R2)
3240 035210 022702 172200      CMP     #SIPDR0,R2
3241 035214 103774      BLO     20$
3242 035216 012637 172516      MOV     (SP)+,@MMR3      ;RESTORE MMR3
3243 035222 012637 177746      MOV     (SP)+,@CONTRL      ;RESTORE CACHE CONTRL
3244 035226 012637 177776      MOV     (SP)+,@PSW      ;RESTORE PSW
3245 035232 012702 177700      MOV     #UDPAR7+2,R2      ;RESTORE PSW THRU MMR0
3246 035236 012642 30$:      MOV     (SP)+,-(R2)
    
```

```
3247 035240 022702 177572      CMP      #MMR0,R2
3248 035244 103774      BLO      30$
3249 035246 012602      MOV      (SP)+,R2      ;RESTORE R2
3250 035250 004737 034446      JSR      PC, TIMIT      ;CHECK THE POWER DOWN
3251 035254 032760 000020 014612      BIT      #SSU,PFRT(R0)  ;SEND SIGNALS?
3252 035262 001403      BEQ      45$           ;NO
3253 035264 012737 000001 014642      MOV      #1, SIGNAL
3254 035272 012737 000001 014712 45$:      MOV      #1,SYNC.1      ;THIS MAY UNLOCK THE OTHER CPUS
3255 035300 012760 034656 014662      MOV      #SPWRDN,PWRTAB(R0) ;SET VECTOR FOR POWER FAIL
3256 035306 005703      TST      R3           ;IS R3 ZERO?
3257 035310 001401      BEQ      50$           ;YES
3258 035312 010316      MOV      R3,(SP)      ;FUDGE RETURN ADDRESS ON STACK
3259 035314 000002      50$:      RTI
3260
3261 035316      $ILLUP:
3262 035316 000000      HALT
3263 035320 000777      BR      .             ;POWER UP BEFORE POWER DOWN COMPLETE
3264                                     ;HANG UP THE PROCESSOR
3265 035322      ILLDN:
3266 035322 000000      HALT
3267 035324 000777      BR      .             ;POWER DOWN BEFORE UP COMPLETE
3268
3269
3270
3271
3272      .SBTTL  END OF PASS ROUTINE
3273
3274      ;*****
3275      ;*INCREMENT THE PASS NUMBER ($PASS)
3276      ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3277      ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT
3278      ;*          CPU #0 > AAAAA
3279      ;*          CPU #1 > BBBBB
3280      ;*          CPU #2 > CCCCC
3281      ;*          CPU #3 > DDDDD
3282      ;*          TOTAL SYSTEM-WIDE ERRORS YYYYY'
3283      ;*WHERE XXXXX,AAAAA,BBBBB,CCCCC,DDDDD, AND YYYYY ARE DECIMAL NUMBERS
3284      ;*IF THERES A MONITOR GO TO IT
3285      ;*IF THERE ISN'T JUMP TO RESTAB
3286
3287 035326      $EOP:
3288 035326 106237 016776      ASRB     SYNC.3      ;;CONTROL ENTRY
3289 035332 103375      BCC      $EOP
3290 035334 105737 016737      TSTB     MPF         ;;MP MODE?
3291 035340 001416      BEQ      4$         ;;BRANCH IF NO
3292 035342 005700      TST      R0         ;;IS THIS THE MASTER?
3293 035344 001414      BEQ      4$         ;;YES
3294 035346 005237 014134      INC      $EOPSG
3295 035352 012737 000001 016776      MOV      #1, SYNC.3 ;;ALLOW ANOTHER CPU IN
3296 035360 005737 014134 1$:      TST      $EOPSG     ;;IS THE MASTER FINISHED?
3297 035364 001375      BNE      1$         ;;BRANCH IF NO
3298 035366 016006 014200      MOV      $$STP(R0),SP ;;RESET THE STACK
3299 035372 000170 037064      JMP      @RESTAB(R0)
3300 035376
3301 035376 005237 014134      INC      $EOPSG
3302 035402 012737 000001 016776      MOV      #1, SYNC.3 ;;ALLOW ANOTHER CPU IN
```

```

3303 035410 023737 014134 016774 5$:   CMP      $EOPSG, CPUACT
3304 035416 001374                BNE      5$
3305 035420 005237 014344                INC      $PASS          ;; INCREMENT THE PASS NUMBER
3306 035424 042737 100000 014344                BIC      #100000,$PASS  ;; DON'T ALLOW A NEG. NUMBER
3307 035432 005327                DEC      (PC)+          ;; LOOP?
3308 035434 000001                $EOPCT: .WORD      1
3309 035436 003402                BLE      1$            ;; YES
3310 035440 000137 036122                JMP      $DOAGN
3311 035444 012737                1$:     MOV      (PC)+,@(PC)+  ;; RESTORE COUNTER
3312 035446 000001                $ENDCT: .WORD      1
3313 035450 035434                $EOPCT
3314 035452 104401 035460                TYPE    ,65$          ;; TYPE ASCIZ STRING
3315 035456 000407                BR      64$          ;; GET OVER THE ASCIZ
3316                ;;65$: .ASCIZ <12><15>/END PASS #/
3317 035476                64$:
3318 035476 013746 014344                MOV      $PASS,-(SP)  ;; SAVE $PASS FOR TYPEOUT
3319                ;; TYPE PASS NUMBER
3320 035502 104405                TYPDS   ;; GO TYPE--DECIMAL ASCII WITH SIGN
3321 035504 104401 035512                TYPE    ,67$          ;; TYPE ASCIZ STRING
3322 035510 000421                BR      66$          ;; GET OVER THE ASCIZ
3323                ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
3324 035554                66$:
3325 035554 105737 016737                TSTB   MPF
3326 035560 001524                BEG     UNIEOP
3327 035562 104401 014333                TYPE    , $CRLF
3328 035566 104401 035574                TYPE    ,69$          ;; TYPE ASCIZ STRING
3329 035572 000404                BR      68$          ;; GET OVER THE ASCIZ
3330                ;;69$: .ASCIZ /CPU#0 /<76>
3331 035604                68$:
3332 035604 005003                CLR     R3
3333 035606 004737 036226                JSR     PC,EOPCID
3334 035612 016346 014042                MOV     $ERTTL(R3),-(SP)  ;; SAVE $ERTTL(R3) FOR TYPEOUT
3335 035616 104405                TYPDS   ;; GO TYPE--DECIMAL ASCII WITH SIGN
3336 035620 104401 014333                TYPE    , $CRLF
3337 035624 104401 035672                TYPE    ,71$          ;; TYPE ASCIZ STRING
3338 035630 000404                BR      70$          ;; GET OVER THE ASCIZ
3339                ;;71$: .ASCIZ /CPU#1 /<76>
3340 035642                70$:
3341 035642 012703 000001                MOV     #1,R3
3342 035646 004737 036226                JSR     PC,EOPCID
3343 035652 016346 014042                MOV     $ERTTL(R3),-(SP)  ;; SAVE $ERTTL(R3) FOR TYPEOUT
3344 035656 104405                TYPDS   ;; GO TYPE--DECIMAL ASCII WITH SIGN
3345 035660 104401 014333                TYPE    , $CRLF
3346 035664 104401 035672                TYPE    ,73$          ;; TYPE ASCIZ STRING
3347 035670 000404                BR      72$          ;; GET OVER THE ASCIZ
3348                ;;73$: .ASCIZ /CPU#2 /<76>
3349 035702                72$:
3350 035702 012703 000002                MOV     #2,R3
3351 035706 004737 036226                JSR     PC,EOPCID
3352 035712 016346 014042                MOV     $ERTTL(R3),-(SP)  ;; SAVE $ERTTL(R3) FOR TYPEOUT
3353 035716 104405                TYPDS   ;; GO TYPE--DECIMAL ASCII WITH SIGN
3354 035720 104401 014333                TYPE    , $CRLF
3355 035724 104401 035732                TYPE    ,75$          ;; TYPE ASCIZ STRING
3356 035730 000404                BR      74$          ;; GET OVER THE ASCIZ
3357                ;;75$: .ASCIZ /CPU#3 /<76>
3358 035742                74$:
  
```

```

3359 035742 012703 000003      MOV      #3,R3
3360 035746 004737 036226      JSR      PC,EOPCID
3361 035752 016346 014042      MOV      $ERTTL(R3),-(SP)      ;;SAVE $ERTTL(R3) FOR TYPEOUT
3362 035756 104405                TYPDS                ;;GO TYPE--DECIMAL ASCII WITH SIGN
3363 035760 104401 014333      TYPE      ,SCLRF
3364 035764 104401 035772      TYPE      ,77$          ;;TYPE ASCII STRING
3365 035770 000420                BR        76$          ;;GET OVER THE ASCII
3366                                ;;77$: .ASCIZ /TOTAL SYSTEMWIDE ERROR COUNT = /
3367 036032                                76$:
3368 036032                                UNIEOP:
3369 036032 013746 014224      MOV      $ERGBL,-(SP)      ;;SAVE $ERGBL FOR TYPEOUT
3370 036036 104405                TYPDS                ;;GO TYPE--DECIMAL ASCII WITH SIGN
3371 036040 104401 014333      TYPE      ,SCLRF
3372 036044 005037 014224      CLR      $ERGBL
3373 036050 012703 014042      MOV      # $ERTTL,R3      ;CLEAR THE
3374 036054 005023                CLR      (R3)+          ;ERROR TOTALS.
3375 036056 005023                CLR      (R3)+
3376 036060 005023                CLR      (R3)+
3377 036062 005023                CLR      (R3)+
3378 036064 005013                CLR      (R3)
3379 036066 000400                BR        99$          ;SKIP OVER IN SUBROUTINE
3380 036070 000404                99$: BR        $GET
3381 036072 013702 000042      $GET42: MOV      @#42,R2      ;;GET MONITOR ADDRESS
3382 036076 001411                BEQ      $DOAGN          ;;BRANCH IF NO MONITOR
3383 036100 000001                WAIT
3384 036102 013702 000042      $GET:  MOV      @#42,R2      ;;INSURE R2 CONTAINS THE MONITORS
3385 036106 001405                BEQ      $DOAGN          ;;RETURN ADDRESS
3386 036110 000005                RESET          ;;CLEAR THE WORLD
3387 036112 004712                $ENDAD: JSR      PC,(R2)   ;;GO TO MONITOR
3388 036114 000240                NOP          ;;SAVE ROOM
3389 036116 000240                NOP          ;;FOR
3390 036120 000240                NOP          ;;ACT11
3391 036122 013746 014660      $DOAGN: MOV      $PSWR, -(SP)
3392 036126 013746 014716      MOV      TYPQUE, -(SP)
3393 036132 013746 014720      MOV      TYPQUE+2, -(SP)
3394 036136 013746 016740      MOV      UBEF, -(SP)
3395 036142 013746 016774      MOV      CPUACT, -(SP)
3396 036146 013746 016736      MOV      FLAGB, -(SP)
3397 036152 004737 020000      JSR      PC, RESTRT
3398 036156 012637 016736      MOV      (SP)+, FLAGB
3399 036162 012637 016774      MOV      (SP)+, CPUACT
3400 036166 012637 016740      MOV      (SP)+, UBEF
3401 036172 012637 014720      MOV      (SP)+, TYPQUE+2
3402 036176 012637 014716      MOV      (SP)+, TYPQUE
3403 036202 012637 014660      MOV      (SP)+, $PSWR
3404 036206 005037 014134      CLR      $EOPSG          ;;CLEAR THE COUNT AND FREE SLAVES
3405 036212 016006 014200      MOV      $$STP(R0),SP    ;;RESET THE STACK
3406 036216 000170 037064      JMP      @RESTAB(R0)    ;RETURN
3407 036222                377 377      000 $ENULL: .BYTE      -1,-1,0      ;;NULL CHARACTER STRING
3408                                036226
3409                                EOPCID:
3410 036226 005002                CLR      R2          ;RESET FOR COUNT
3411 036230 026203 014226      65$:  CMP      $CPUID(R2),R3    ;SID MATCH?
3412 036234 001404                BEQ      64$
3413 036236 005722                TST      (R2)+
3414 036240 020227 000010      CMP      R2,#10          ;INCREMENT R2 BY 2

```

M 6

```

3415 036244 002771          BLT      65$
3416 036246 010203      64$:  MOV      R2,R3          ;MOV LOGICAL ID TO 2ND OPERAND
3417 036250 000207          RTS      PC
3418
3419
3420
3421
3422          .SBTTL  SCOPE HANDLER ROUTINE
3423
3424          ;*****
3425          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3426          ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
3427          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3428          ;*SW14=1      LOOP ON TEST
3429          ;*SW09=1      LOOP ON ERROR
3430          ;*CALL
3431          ;*          SCOPE          ;;SCOPE=IOT
3432
3433          $SCOPE:
3434 036252 032770 040000 014160 1$:  BIT      #BIT14,@SWR(R0)  ;;LOOP ON PRESENT TEST?
3435 036260 001056          BNE      $OVER          ;;YES IF SW14=1
3436          ;*****START OF CODE FOR THE XOR TESTER*****
3437 036262 000416          $XTSTR: BR      6$          ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
3438          ;THIS INSTRUCTION TO A 'NOP' (NOP=240)
3439 036264 013746 000004          MOV      @WERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3440 036270 012737 036310 000004          MOV      #5$,@WERRVEC  ;;SET FOR TIMEOUT
3441 036276 005737 177060          TST      @#177060      ;;TIME OUT ON XOR?
3442 036302 012637 000004          MOV      (SP)+,@WERRVEC ;;RESTORE THE ERROR VECTOR
3443 036306 000423          BR      $SVLAD         ;;GO TO THE NEXT TEST
3444 036310 022626          5$:  CMP      (SP)+,(SP)+  ;;CLEAR THE STACK AFTER A TIME OUT
3445 036312 012637 000004          MOV      (SP)+,@WERRVEC ;;RESTORE THE ERROR VECTOR
3446 036316 000411          BR      7$          ;;LOOP ON THE PRESENT TEST
3447 036320          6$:  ;*****END OF CODE FOR THE XOR TESTER*****
3448 036320 105761 014006          2$:  TSTB   $ERFLG(R1)    ;;HAS AN ERROR OCCURRED?
3449 036324 001414          BEQ      $SVLAD         ;;BR IF NO
3450 036326 032770 001000 014160          BIT      #BIT09,@SWR(R0) ;;LOOP ON ERROR?
3451 036334 001002          BNE      7$          ;;BR IF NO
3452 036336 000160 036352          JMP      4$(R0)
3453 036342 013760 014032 014022 7$:  MOV      $LPERR,$LPADR(R0) ;;SET LOOP ADDRESS TO LAST SCOPE
3454 036350 000422          BR      $OVER
3455 036352 105061 014006          4$:  CLRB   $ERFLG(R1)    ;;ZERO THE ERROR FLAG
3456 036356 105261 014002          $SVLAD: INCB  $STNM(R1)  ;;COUNT TEST NUMBERS
3457 036362 005710          TST      (R0)          ;;IS THIS THE MASTER
3458 036364 001003          BNE      1$          ;;NO,
3459 036366 116137 014002 014342          1$:  MOVB   $STNM(R1),$TESTN ;;SET TEST NUMBER IN APT MAILBOX
3460 036374
3461 036374 011660 014022          MOV      (SP),$LPADR(R0) ;;SAVE SCOPE LOOP ADDRESS
3462 036400 011660 014032          MOV      (SP),$LPERR(R0) ;;SAVE ERROR LOOP ADDRESS
3463 036404 005060 014322          CLR      $ESCAPE(R0)    ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3464 036410 112737 000001 014060          MOVB   #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3465 036416 113771 014002 014170 $OVER: MOVB   $STNM,@DISPLAY(R1) ;;DISPLAY TEST NUMBER
3466 036424 016016 014022          MOV      $LPADR(R0),(SP) ;;FUDGE RETURN ADDRESS
3467 036430 000002          RTI
3468
3469
3470
    
```

```

3471
3472          .SBTTL  ERROR HANDLER ROUTINE
3473
3474          ::*****
3475          ::*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3476          ::*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3477          ::*MADE BY THE FAILING PROCESSOR
3478          ::*AND TYPE OUT THE PROCESSOR ID AND PC OF THE ERROR INSTRUCTION
3479          ::*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3480          ::*SW15=1      HALT ON ERROR
3481          ::*SW09=1      LOOP ON ERROR
3482          ::*CALL
3483          ::*      ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3484
3485          $ERROR:
3486          036432 105261 014006 7$:      INCB      $ERFLG(R1)      ;;SET THE ERROR FLAG
3487          036436 001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
3488          036440 116170 014002 014170  MOVB      $STNM(R1),@DISPLAY(R0) ;;DISPLAY TEST NUMBER
3489          036446 005260 014042      INC      $ERTTL(R0)      ;;INC THE ERROR COUNT
3490          036452 005237 014224      INC      $ERGBL
3491          036456 011660 014064      MOV      (SP), $ERRPC(R0) ;;GET ADDRESS OF ERROR INSTRUCTION
3492          036462 162760 000002 014064  SUB      #2, $ERRPC(R0)
3493          036470 117061 014064 014054  MOVB      @ $ERRPC(R0), $ITEMB(R1) ;;STRIP AND SAVE THE ERROR ITEM CODE
3494          036476 010660 014074      MOV      SP, $ERRSP(R0) ;;SAVE THE CURRENT STACK POINTER
3495          036502 005770 014160      TST      @ $R(R0)      ;;HALT ON ERROR?
3496          036506 100025      BPL      10$      ;;SKIP IF CONTINUE
3497          036510 005700      TST      R0      ;;IS THIS THE MASTER?
3498          036512 001022      BNE      3$      ;;NO
3499          036514 104401 036522      TYPE      ,70$      ;;TYPE ASCIZ STRING
3500          036520 000417      BR      75$      ;;GET OVER THE ASCIZ
3501          036522 005015 040510 052114 70$:      .ASCIZ <15><12>/HALT ON MASTER IN $ERROR/<15><12>
3502          036530 047440 020116 040515
3503          036536 052123 051105 044440
3504          036544 020116 042444 051122
3505          036552 051117 005015      000
3506          036560      .EVEN
3507          036560      75$:
3508          036560 000000      3$:      HALT
3509          036562 000413      10$:      BR      6$      ;;NO LOOP ON ERROR
3510          036564 032770 001000 014160 4$:      BIT      #BIT9, @SWR(R0) ;;LOOP ON ERROR SWITCH SET?
3511          001402      BEQ      5$      ;;BR IF NO
3512          016016 014032      MOV      $LPERR(R0), (SP) ;;FUDGE RETURN ADDRESS
3513          036600 005760 014322      5$:      TST      $ESCAPE(R0) ;;CHECK FOR AN ESCAPE ADDRESS
3514          036604 001402      BEQ      6$      ;;BR IF NONE
3515          036606 016016 014322      MOV      $ESCAPE(R0), (SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
3516
3517          036612      6$:
3518          036612 122737 000001 014356      CMPB      #APTENV, $ENV      ;;RUNNING IN APT MODE
3519          036620 001007      BNE      11$      ;;NO, SKIP APT ERROR REPORT
3520          036622 116137 014054 036634      MOVB      $ITEMB(R1), 21$      ;;SET ITEM NUMBER AS ERROR NUMBER
3521          036630 004737 037112      JSR      PC, $ATY4      ;;REPORT FATAL ERROR TO APT
3522          036634      000      21$:      .BYTE      0
3523          036635      000      .BYTE      0
3524          036636 000777      22$:      BR      22$      ;;APT ERROR LOOP
3525          036640      11$:
3526          036640 022737 036112 000042      CMP      #SENDAD, @#42      ;;ACT-11 AUTO-ACCEPT?
    
```

```

3527 036646 001001          BNE      12$          ;;BRANCH IF NO
3528 036650 000000          HALT
3529 036652          12$:
3530 036652          $ERRTYP:
3531
3532 036652 106237 017002    64$:  ASRB      ERRLCK
3533 036656 103375          BCC      64$
3534 036660 010037 017064    MOV      R0,      TYPLCK          ;ALLOW THIS CPU TO ENTER TYPE ROUTINE
3535 036664 005060 017012    CLR      NOPRMP(R0)          ;;PRINT MESSAGES WITH PROMPTS
3536 036670 116105 014054    MOV      $ITEMB(R1),R5          ;;GET THE ERROR ITEM CODE
3537 036674 072527 000003    ASH      #3,      R5
3538 036700 162705 000010    SUB      #10,     R5
3539 036704 062705 017070    ADD      #$ERRTB,R5          ;;ADD THE ADDR. OF THE ERROR TABLE
3540 036710 011537 036716    MOV      (R5),1$          ;;SET UP TO...
3541 036714 1044J1          TYPE
3542 036716 000000          1$:  .WORD      0          ;;TYPE THE ERROR HEADER
3543 036720 104401 014333    TYPE      ,SCLF
3544 036724 005725          TST      (R5)+          ;;INCREMENT R5 BY 2
3545 036726 005715          TST      (R5)          ;;IS THERE A DATA HEADER?
3546 036730 001406          BEQ      10$          ;;BRANCH IF NO
3547 036732 011537 036740    MOV      (R5),2$
3548 036736 104401          TYPE
3549 036740 000000          2$:  .WORD      0
3550 036742 104401 014333    TYPE      ,SCLF
3551 036746 005725          10$: TST      (R5)+          ;;INCREMENT R5
3552 036750 005715          TST      (R5)          ;;IS THERE DATA TO BE TYPED?
3553 036752 001433          BEQ      20$
3554 036754 011505          MOV      (R5),R5          ;;GET THE DATA TABLE ADDRESS
3555 036756 005715          15$: TST      (R5)          ;;ARE WE AT THE END OF THE DATA TABLE?
3556 036760 001430          BEQ      20$          ;;BRANCH IF YES
3557 036762 000240          NOP
3558 036764 011537 014524    MOV      (R5),   YYY          ;POINT TO THE LOCATION WITH THE NUMBER
3559 036770 022527 014002    CMP      (R5)+,  #$STSTM
3560 036774 001405          BEQ      16$
3561 036776 060037 014524    ADD      R0,     YYY
3562 037002 017746 155516    MOV      @YYY,  -(SP)
3563 037006 000405          BR       17$
3564 037010 005046          16$: CLR      -(SP)
3565 037012 060137 014524    ADD      R1,     YYY
3566 037016 117716 155502    MOV      @YYY,  (SP)
3567 037022 104402          17$: TYPOC
3568 037024 000240          NOP
3569 037026 104401 037034    TYPE      ,66$          ;;TYPE ASCIZ STRING
3570 037032 000402          BR       65$          ;;GET OVER THE ASCIZ
3571          ;;66$: .ASCIZ  / /
3572 037040          65$:
3573 037040 000746          BR       15$
3574 037042 104401 014333    20$: TYPE      ,SCLF
3575 037046 012737 177777 017064    MOV      #-1,    TYPLCK          ;ALLOW ENTRY INTO TYPE ROUTINE
3576 037054 012737 000001 017002    MOV      #1,ERRLCK
3577
3578 037062 000002          RESTAB: RTI          ;;RETURN
3579 037064 021450          TST1          ;;END OF PASS RETURN
3580 037066 021450          TST1
3581 037070 021450          TST1
3582 037072 021450          TST1

```


3583
3584
3585
3586
3587
3588
3589
3590
3591
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638

.SBTTL APT COMMUNICATIONS ROUTINE

037074 112737 000001 037340
037102 112737 000001 037336
037110 000403
037112 112737 000001 037340
037120
037120 010046
037122 010146
037124 105737 037336
037130 001450
037132 122737 000001 014356
037140 001031
037142 132737 000100 014357
037150 001425
037152 017600 000004
037156 062766 000002 000004
037164 005737 014336
037170 001375
037172 010037 014352
037176 105720
037200 001376
037202 163700 014352
037206 006200
037210 010037 014354
037214 012737 000004 014336
037222 000413
037224 017637 000004 037250
037232 062766 000002 000004
037240 013746 177776
037244 004737 040072
037250 000000
037252
037252 105737 037340
037256 001416
037260 005737 014356
037264 001413
037266 005737 014336
037272 001375
037274 017637 000004 014340
037302 062766 000002 000004
037310 005237 014336
037314 105037 037340
037320 105037 037337
037324 105037 037336
037330 012601

```
*****  
SATY1:  MOVB  #1,$FFLG      ;;TO REPORT FATAL ERROR  
SATY3:  MOVB  #1,$MFLG      ;;TO TYPE A MESSAGE  
BR      SATYC  
SATY4:  MOVB  #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR  
SATYC:  
MOV     R0,-(SP)           ;;PUSH R0 ON STACK  
MOV     R1,-(SP)           ;;PUSH R1 ON STACK  
TSTB   $MFLG              ;;SHOULD TYPE A MESSAGE?  
BEQ    5$                  ;;IF NOT: BR  
CMPB   #APTENV,$ENV       ;;OPERATING UNDER APT?  
BNE    3$                  ;;IF NOT: BR  
BITB   #APTPOOL,$ENVM     ;;SHOULD SPOOL MESSAGES?  
BEQ    3$                  ;;IF NOT: BR  
MOV     @4(SP),R0          ;;GET MESSAGE ADDR.  
ADD     #2,4(SP)           ;;BUMP RETURN ADDR.  
TST    $MSGTYPE           ;;SEE IF DONE W/ LAST XMISSION?  
BNE    1$                  ;;IF NOT: WAIT  
MOV     R0,$MSGAD         ;;PUT ADDR IN MAILBOX  
TSTB   (R0)+              ;;FIND END OF MESSAGE  
BNE    2$  
SUB     $MSGAD,R0          ;;SUB START OF MESSAGE  
ASR    R0                  ;;GET MESSAGE LNGTH IN WORDS  
MOV     R0,$MSGGLT        ;;PUT LENGTH IN MAILBOX  
MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.  
BR     5$  
3$:    MOV     @4(SP),4$    ;;PUT MSG ADDR IN JSR LINKAGE  
ADD     #2,4(SP)           ;;BUMP RETURN ADDRESS  
MOV     177776,-(SP)      ;;PUSH 177776 ON STACK  
JSR    PC,$TYPE           ;;CALL TYPE MACRO  
4$:    .WORD  0  
5$:  
10$:   TSTB   $FFLG        ;;SHOULD REPORT FATAL ERROR?  
BEQ    12$                ;;IF NOT: BR  
TST    $ENV               ;;RUNNING UNDER APT?  
BEQ    12$                ;;IF NOT: BR  
11$:   TST    $MSGTYPE     ;;FINISHED LAST MESSAGE?  
BNE    11$                ;;IF NOT: WAIT  
MOV     @4(SP),$FATAL     ;;GET ERROR #  
ADD     #2,4(SP)           ;;BUMP RETURN ADDR.  
INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR  
12$:   CLRB   $FFLG        ;;CLEAR FATAL FLAG  
CLRB   $LFLG              ;;CLEAR LOG FLAG  
CLRB   $MFLG              ;;CLEAR MESSAGE FLAG  
MOV     (SP)+,R1          ;;POP STACK INTO R1
```

3639 037332 012600
3640 037334 000207
3641 037336 000
3642 037337 000
3643 037340 000
3644 037342
3645 000200
3646 000001
3647 000100
3648 000040
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678 037342 017646 000000
3679 037346 116637 000001 037635
3680 037354 112637 037637
3681 037360 062716 000002
3682 037364 000406
3683 037366 112737 000001 037635
3684 037374 112737 000006 037637
3685 037402 112737 000005 037634
3686 037410 010346
3687 037412 010446
3688 037414 010546
3689 037416 113737 037637 037640
3690 037424 005437 037640
3691 037430 062737 000006 037640
3692 037436 113737 037640 037636
3693 037444 113737 037635 037640
3694 037452 016605 000012

```
MOV (SP)+,R0      ;;POP STACK INTO R0
RTS PC            ;;RETURN
$MFLG: .BYTE 0    ;;MESSG. FLAG
$LFLG: .BYTE 0    ;;LOG FLAG
$FFLG: .BYTE 0    ;;FATAL FLAG
          .EVEN
APTSIZE=200
APTENV=001
APTPOOL=100
APTCSUP=040

.SBttl BINARY TO OCTAL (ASCII) AND TYPE
*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV   NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOS                ;;CALL FOR TYPEOUT
*   .BYTE N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*   MOV   NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON                ;;CALL FOR TYPEOUT
*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV   NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOC                ;;CALL FOR TYPEOUT
$TYPOS: MOV @ (SP),-(SP)  ;;PICKUP THE MODE
        MOVB 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH
        MOVB (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE
        ADD #2,(SP)      ;;ADJUST RETURN ADDRESS
        BR $TYPON
$TYPOC: MOVB #1, $OFILL  ;;SET THE ZERO FILL SWITCH
        MOVB #6, $OMODE+1 ;;SET FOR SIX(6) DIGITS
$TYPON: MOVB #5, $OCNT   ;;SET THE ITERATION COUNT
        MOV R3,-(SP)    ;;SAVE R3
        MOV R4,-(SP)    ;;SAVE R4
        MOV R5,-(SP)    ;;SAVE R5
        MOVB $OMODE+1,DIGITS ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG DIGITS      ;;SUBTRACT IT FOR MAX. ALLOWED
        ADD #6,DIGITS
        MOVB DIGITS,$OMODE ;;SAVE IT FOR USE
        MOVB $OFILL,DIGITS ;;GET THE ZERO FILL SWITCH
        MOV 12(SP),R5    ;;PICKUP THE INPUT NUMBER
```

```

3695 037456 005003          CLR      R3          ;;CLEAR THE OUTPUT WORD
3696 037460 006105          1$:    ROL      R5          ;;ROTATE MSB INTO 'C'
3697 037462 000404          BR       3$          ;;GO DO MSB
3698 037464 006105          2$:    ROL      R5          ;;FORM THIS DIGIT
3699 037466 006105          ROL      R5
3700 037470 006105          ROL      R5
3701 037472 010503          MOV      R5,R3
3702 037474 006103          3$:    ROL      R3          ;;GET LSB OF THIS DIGIT
3703 037476 105337 037636  DECB     $OMODE       ;;TYPE THIS DIGIT?
3704 037502 100034          BPL      7$          ;;BR IF NO
3705 037504 042703 177770  BIC      #177770,R3   ;;GET RID OF JUNK
3706 037510 001003          BNE      4$          ;;TEST FOR 0
3707 037512 005737 037640  TST      DIGITS       ;;SUPPRESS THIS 0?
3708 037516 001404          BEQ      5$          ;;BR IF YES
3709 037520 005237 037640  4$:    INC      DIGITS   ;;DON'T SUPPRESS ANYMORE 0'S
3710 037524 052703 000060  BIS      #'0,R3       ;;MAKE THIS DIGIT ASCII
3711 037530 052703 000040  5$:    BIS      #' ,R3   ;;MAKE ASCII IF NOT ALREADY
3712 037534 013704 037546  MOV      9$, R4       ;;POINT TO ERRBUF TABLE
3713 037540 110324          MOVB     R3, (R4)+    ;;PUT THE CHARACTER IN THE TABLE
3714 037542 105014          CLRB     (R4)        ;;MAKE IT A MINI-ASCII MSG
3715 037544 104401          TYPE
3716 037546 047266          9$:    ERRBUF        ;;HERE IS THE LOCATION OF MSG
3717 037550 062737 000002 037546  ADD      #2, 9$       ;;MOVE TO NEXT TABLE LOC
3718 037556 023727 037546 111266  CMP      9$, #END     ;;AT THE END OF ALLOWED BUF AREA?
3719 037564 002403          BLT      7$          ;;BRANCH IF NO
3720 037566 012737 047266 037546  MOV      #ERRBUF,9$   ;;ELSE POINT TO BEGINNING AGAIN
3721 037574 105337 037634          7$:    DECB     $OCNT   ;;COUNT BY 1
3722 037600 003331          BGT      2$          ;;BR IF MORE TO DO
3723 037602 002403          BLT      6$          ;;BR IF DONE
3724 037604 005237 037640          INC      DIGITS       ;;INSURE LAST DIGIT ISN'T A BLANK
3725 037610 000725          BR       2$          ;;GO DO THE LAST DIGIT
3726 037612 012605          6$:    MOV      (SP)+,R5  ;;RESTORE R5
3727 037614 012604          MOV      (SP)+,R4     ;;RESTORE R4
3728 037616 012603          MOV      (SP)+,R3     ;;RESTORE R3
3729 037620 016666 000002 000004  MOV      2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
3730 037626 012616          MOV      (SP)+,(SP)
3731 037630 000002          RTI
3732 037632 000          8$:    .BYTE 0          ;;RETURN
3733 037633 000          .BYTE 0          ;;STORAGE FOR ASCII DIGIT
3734 037634 000          $OCNT: .BYTE 0      ;;TERMINATOR FOR TYPE ROUTINE
3735 037635 000          $OFILL: .BYTE 0     ;;OCTAL DIGIT COUNTER
3736 037636 000000          $OMODE: .WORD 0     ;;ZERO FILL SWITCH
3737 037640 000000          DIGITS: .WORD 0    ;;NUMBER OF DIGITS TO TYPE

```

3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

;*****
;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;*REPLACED WITH SPACES.
;*CALL:

```

```

3751      :*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
3752      :*      TYPDS      ;;GO TO THE ROUTINE
3753
3754      $TYPDS:
3755      037642  010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
3756      037644  010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
3757      037646  010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
3758      037650  010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
3759      037652  010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
3760      037654  012746  020200      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
3761      037660  016605  000020      MOV      20(SP),R5    ;;GET THE INPUT NUMBER
3762      037664  100004      BPL      1$           ;;BR IF INPUT IS POS.
3763      037666  005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
3764      037670  112766  000055  000001      MOVB     #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
3765      037676  005000      1$:      CLR      R0           ;;ZERO THE CONSTANTS INDEX
3766      037700  012703  040062      MOV      #SDBLK,R3    ;;SETUP THE OUTPUT POINTER
3767      037704  112723  000040      MOVB     #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
3768      037710  005002      2$:      CLR      R2           ;;CLEAR THE BCD NUMBER
3769      037712  016001  040052      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
3770      037716  160105      3$:      SUB      R1,R5        ;;FORM THIS BCD DIGIT
3771      037720  002402      BLT      4$           ;;BR IF DONE
3772      037722  005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
3773      037724  000774      BR       3$
3774      037726  060105      4$:      ADD      R1,R5        ;;ADD BACK THE CONSTANT
3775      037730  005702      TST      R2           ;;CHECK IF BCD DIGIT=0
3776      037732  001002      BNE      5$           ;;FALL THROUGH IF 0
3777      037734  105716      TSTB     (SP)         ;;STILL DOING LEADING 0'S?
3778      037736  100407      BMI      7$           ;;BR IF YES
3779      037740  106316      5$:      ASLB     (SP)         ;;MSD?
3780      037742  103003      BCC      6$           ;;BR IF NO
3781      037744  116663  000001  177777      MOVB     1(SP),-1(R3) ;;YES--SET THE SIGN
3782      037752  052702  000060      BIS      #'0,R2      ;;MAKE THE BCD DIGIT ASCII
3783      037756  052702  000040      6$:      BIS      #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
3784      037762  110223      7$:      MOVB     R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
3785      037764  005720      TST      (R0)+       ;;JUST INCREMENTING
3786      037766  020027  000010      CMP      R0,#10     ;;CHECK THE TABLE INDEX
3787      037772  002746      BLT      2$           ;;GO DO THE NEXT DIGIT
3788      037774  003002      BGT      8$           ;;GO TO EXIT
3789      037776  010502      MOV      R5,R2       ;;GET THE LSD
3790      040000  000764      BR       6$           ;;GO CHANGE TO ASCII
3791      040002  105726      8$:      TSTB     (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?
3792      040004  100003      BPL      9$           ;;BR IF NO
3793      040006  116663  177777  177776      MOVB     -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
3794      040014  105013      9$:      CLRB     (R3)        ;;SET THE TERMINATOR
3795      040016  012605      MOV      (SP)+,R5    ;;POP STACK INTO R5
3796      040020  012603      MOV      (SP)+,R3    ;;POP STACK INTO R3
3797      040022  012602      MOV      (SP)+,R2    ;;POP STACK INTO R2
3798      040024  012601      MOV      (SP)+,R1    ;;POP STACK INTO R1
3799      040026  012600      MOV      (SP)+,R0    ;;POP STACK INTO R0
3800      040030  000240      NOP
3801      040032  104401  040062      TYPE     ,SDBLK      ;;NOW TYPE THE NUMBER
3802      040036  000240      NOP
3803      040040  016666  000002  000004      MOV      2(SP),4(SP)  ;;ADJUST THE STACK
3804      040046  012616      MOV      (SP)+,(SP)
3805      040050  000002      RTI
3806      040052  023420      $DTBL:  10000.      ;;RETURN TO USER

```

```

3807 040054 001750          1000.
3808 040056 000144          100.
3809 040060 000012          10.
3810 040062 000004          $DBLK: .BLKW 4
3811
3812
3813
3814
3815          .SBTTL TYPE SERVICE
3816 040072          $TYPE:
3817 040072 132737 000040 014357 BITB #40, $ENVM ;INHIBIT PRINT OUT?
3818 040100 001403          BEQ 6$ ;BRANCH IF NO
3819 040102 062716 000002 ADD #2, (SP) ;SET RETURN
3820 040106 000002          RTI
3821 040110          6$:
3822 040110 005737 017002 TST ERRLCK ;IS A CPU IN THE ERROR ROUTINE?
3823 040114 001003          BNE 3$ ;BRANCH IF NO
3824 040116 020037 017064 CMP R0, TYPLCK ;IS THIS CPU FROM THE ERROR ROUTINE?
3825 040122 001363          BNE $TYPE ;BRANCH IF NO
3826 040124 010246          3$: MOV R2, -(SP) ;STORE REGISTERS USED IN THIS PROGRAM
3827 040126 010346          MOV R3, -(SP)
3828 040130 010446          MOV R4, -(SP)
3829 040132 010546          MOV R5, -(SP)
3830 040134 105737 016737 TSTB MPF
3831 040140 001003          BNE 1$
3832 040142 012704 000400 MOV #400, R4 ;DONT TYPE A CPUID
3833 040146 000414          BR 2$
3834 040150 017705 156654 1$: MOV @ACR, R5 ;PUT SELF ID INTO R5
3835 040154 072527 177770 ASH #-10, R5
3836 040160 010504          MOV R5, R4 ;PUT ID INTO R4
3837 040162 016002 017012 MOV NOPRMP(R0),R2 ;COPY THE PROMPT FLAG
3838 040166 000302          SWAB R2 ;GET IT INTO THE LEFT HALF
3839 040170 050204          BIS R2,R4 ;PUT IT IN R5
3840 040172 012760 000001 017012 MOV #1, NOPRMP(R0)
3841 040200 106237 017024 2$: ASRB TQL1 ;ENTER TYPQUE CRITICAL SECTION
3842 040204 103375          BCC 2$
3843          ::***** ENQUE *****
3844 040206 013703 014720 MOV TYPQUE+2,R3 ;COPY REAR INDEX
3845 040212 005723          TST (R3)+ ;INCREMENT BY 2
3846 040214 010413          64$: MOV R4,(R3) ;QUEUE THE ELEMENT
3847 040216 010337 014720 MOV R3,TYPQUE+2 ;UPDATE THE REAR INDEX
3848          ::***** ENQUE *****
3849 040222          4$:
3850          ::***** ENQUE *****
3851 040222 013703 014720 MOV TYPQUE+2,R3 ;COPY REAR INDEX
3852 040226 005723          TST (R3)+ ;INCREMENT BY 2
3853 040230 016613 000010 65$: MOV 10(SP),(R3) ;QUEUE THE ELEMENT
3854 040234 010337 014720 MOV R3,TYPQUE+2 ;UPDATE THE REAR INDEX
3855          ::***** ENQUE *****
3856 040240 013343          MOV @R3+, -(R3) ;GET PC OF MSG
3857 040242 012737 000001 017024 5$: MOV #1,TQL1 ;CLEAR CRITICAL SECTION
3858 040250 005700          TST R0 ;IS THIS THE MASTER?
3859 040252 001407          BEQ 11$ ;BRANCH IF YES.
3860 040254 012605          MOV (SP)+, R5 ;RESTORE THE REGISTERS
3861 040256 012604          MOV (SP)+, R4
3862 040260 012603          MOV (SP)+, R3

```

```

3863 040262 012602      MOV    (SP)+, R2
3864 040264 062716 000002  ADD    #2, (SP)
3865 040270 000002      RTI
3866 040272      10$:
3867 040272 106237 017024 11$: ASRB   TQL1      ;ENTER CRITICAL SECTOR
3868 040276 103375      BCC   11$
3869 040300      HERE:
3870
3871 040300 023737 014716 014720  ;;***** DEQUE *****
3872 040306 001456      CMP    TYPQUE,TYPQUE+2 ;;ARE THE INDICIES EQUAL?
3873 040310 013703 014716      BEQ    13$          ;;YES,THE QUEUE IS EMPTY
3874 040314 005723      MOV    TYPQUE,R3    ;;COPY FRONT INDEX INTO R3
3875 040316 011304      TST   (R3)+        ;;INC. R3 BY TWO
3876 040320 010337 014716      MOV    (R3),R4     ;;DEQUEUE AN ELEMENT
3877 040320 010337      MOV    R3,TYPQUE   ;;UPDATE FRONT INDEX
3878 040324 032704 000400      ;;*****
3879 040330 001025      BIT   #BIT8,R4     ;WAS THE NOPRMP(R0) BIT SET?
3880 040332 062704 021060      BNE   12$
3881 040336 112702 000015      ADD   #'0',R4      ;MAKE CPUID A CHARACTER
3882 040342 004737 040504      MOVB  #CR,R2       ;TYPE # CARRAGE RETURN
3883 040346 112702 000012      JSR   PC,TYPIT     ;TYPE CRLF
3884 040352 004737 040504      MOVB  #LF,R2
3885 040356 110402      JSR   PC,TYPIT     ;TYPE IT
3886 040360 004737 040504      MOVB  R4,R2        ;CPU ID
3887 040364 112702 021076      JSR   PC,TYPIT     ;TYPE IT
3888 040370 004737 040504      MOVB  #'>',R2
3889 040374 112702 021040      JSR   PC,TYPIT     ;TYPE IT
3890 040400 004737 040504      MOVB  #' ',R2
3891 040404      JSR   PC,TYPIT     ;TYPE A SPACE
3892
3893 040404 023737 014716 014720 12$:
3894 040412 001414      ;;***** DEQUE *****
3895 040414 013703 014716      CMP    TYPQUE,TYPQUE+2 ;;ARE THE INDICIES EQUAL?
3896 040420 005723      BEQ    13$          ;;YES,THE QUEUE IS EMPTY
3897 040422 011304      MOV    TYPQUE,R3    ;;COPY FRONT INDEX INTO R3
3898 040424 010337 014716      TST   (R3)+        ;;INC. R3 BY TWO
3899 040424 010337      MOV    (R3),R4     ;;DEQUEUE AN ELEMENT
3900 040430 112402      MOV    R3,TYPQUE   ;;UPDATE FRONT INDEX
3901 040432 005702      ;;*****
3902 040434 001721      MOVB  (R4)+,R2     ;GET A CHARECTER
3903 040436 004737 040504      TST   R2
3904 040442 000772      BEQ   HERE        ;WE'RE DONE ,R2 IS CLEAR
3905 040444 012737 014722 014716 14$: JSR   PC,TYPIT     ;TYPE THE CHARECTER
3906 040452 012737 014722 014720  BR   14$          ;LOOP
3907 040460 012737 000001 017024 13$: MOV   #TYPQUE+4, TYPQUE
3908 040466 012605      MOV   #TYPQUE+4, TYPQUE+2
3909 040470 012604      MOV   #1, TQL1    ;RETURN
3910 040472 012603      MOV   (SP)+, R5   ;RESTORE THE REGISTERS
3911 040474 012602      MOV   (SP)+, R4
3912 040476 062716 000002      MOV   (SP)+, R3
3913 040502 000002      MOV   (SP)+, R2
3914
3915 040504 105777 153504      ADD   #2, (SP)
3916 040510 100375      RTI
3917 040512 110277 153500      TYPIT: TSTB @STPS  ;WAIT UNTIL PRINTER IS READY
3918 040516 000207      BPL  TYPIT
3919 040516 000207      MOVB R2,@STPB   ;TYPE THE CHARACTER
3920 040516 000207      RTS   PC        ;RETURN

```

```

3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931 040520 010260 014552
3932 040524 011602
3933 040526 005742
3934 040530 111202
3935 040532 006302
3936 040534 016202 040566
3937 040540 010260 014562
3938 040544 016002 014552
3939 040550 000170 014562
3940
3941
3942
3943
3944
3945 040554 011646
3946 040556 016666 000004 000002
3947 040564 000002
3948
3949
3950
3951
3952
3953
3954
3955
3956 040566 040554
3957 040570 040072
3958 040572 037366
3959 040574 037342
3960 040576 037402
3961 040600 037642
3962
3963
3964
3965 040602 000170 014672
3966
3967 040606 012737 177777 014602
3968 040614 005037 014602
3969 040620 000177 154036
3970 040624 000170 014702
3971
3972 040630
3973 040630 013760 177766 014250
3974 040636 104005

```

.SBTTL TRAP DECODER

```

:*****
:*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION
:*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
:*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
:*GO TO THAT ROUTINE.

```

```

$TRAP: MOV R2, SAVRG(R0)
1$: MOV (SP),R2 ;:GET TRAP ADDRESS
TST -(R2) ;:BACKUP BY 2
MOVB (R2),R2 ;:GET RIGHT BYTE OF TRAP
ASL R2 ;:POSITION FOR INDEXING
MOV $TRPAD(R2),R2 ;:INDEX TO TABLE
MOV R2, ROUTE(R0)
MOV SAVRG(R0), R2
JMP @ROUTE(R0)

```

::THIS IS USE TO HANDLE THE 'GETPRI' MACRO

```

$TRAP2: MOV (SP),-(SP) ;:MOVE THE PC DOWN
MOV 4(SP),2(SP) ;:MOVE THE PSW DOWN
RTI ;:RESTORE THE PSW

```

.SBTTL TRAP TABLE

```

:*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
:*BY THE 'TRAP' INSTRUCTION.

```

	ROUTINE	
\$TRPAD:	.WORD \$TRAP2	
	\$TYPE ;:CALL=TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
	\$TYPOC ;:CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS ;:CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON ;:CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$TYPDS ;:CALL=TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)

```

ISTDIS: JMP @ISTTAB(R0) ;:IIST INTERRUPT DISPATCHER
PWRDIS: MOV #-1,FLAG ;:FIRST INSTRUCTION FLAG
CLR FLAG ;:NOW CLEAR IT
ERRDIS: JMP @ERRTAB(R0) ;:CPU ERROR DISPATCHER
CPUER: MOV @#CPUERR,$REG1(R0) ;:CPU ERROR REG.
ERROR 5

```

3975	040640	000000				HALT
3976	040642	000002				RTI ;RETURN
3977						.SBTTL DATA AREA
3978	040644	005015	042503	041113	TM1:	.ASCIZ <CR><LF>\CEKBG-C 11/70 POW FAIL\
3979	040652	026507	020103	030440		
3980	040660	027461	030067	050040		
3981	040666	053517	043040	044501		
3982	040674	000114				
3983	040676	005015	053523	052111	TM2:	.ASCIZ <CR><LF>\SWITCH REGISTER = \
3984	040704	044103	051040	043505		
3985	040712	051511	042524	020122		
3986	040720	020075	000			
3987	040723	015	055412	047125	TM4:	.ASCIZ <CR><LF>/[UNIBUS EXERCISER WILL BE USED]/
3988	040730	041111	051525	042440		
3989	040736	042530	041522	051511		
3990	040744	051105	053440	046111		
3991	040752	020114	042502	052440		
3992	040760	042523	056504	000		
3993	040765	015	055412	047125	TM5:	.ASCIZ <CR><LF>/[UNIBUS EXERCISER WILL NOT BE USED]/
3994	040772	041111	051525	042440		
3995	041000	042530	041522	051511		
3996	041006	051105	053440	046111		
3997	041014	020114	047516	020124		
3998	041022	042502	052440	042523		
3999	041030	056504	000			
4000	041033	015	055412	052515	TM6:	.ASCIZ <CR><LF>/[MULTIPROCESSOR MODE IS IN EFFECT]/
4001	041040	052114	050111	047522		
4002	041046	042503	051523	051117		
4003	041054	046440	042117	020105		
4004	041062	051511	044440	020116		
4005	041070	043105	042506	052103		
4006	041076	000135				
4007	041100	005015	052533	044516	TM7:	.ASCIZ <CR><LF>/[UNIPROCESSOR MODE IS IN EFFECT]/
4008	041106	051120	041517	051505		
4009	041114	047523	020122	047515		
4010	041122	042504	044440	020123		
4011	041130	047111	042440	043106		
4012	041136	041505	056524	000		
4013	041143	040	047533	044124	TM10:	.ASCIZ / [OTHER CPUS ARE RUNNING.]/
4014	041150	051105	041440	052520		
4015	041156	020123	051101	020105		
4016	041164	052522	047116	047111		
4017	041172	027107	000135			
4018	041176	005015	041407	052520	TM11:	.ASCIZ <CR><LF><07>/CPU #/
4019	041204	021440	000			
4020	041207	040	050123	041505	TM12:	.ASCIZ / SPECIFIED BUT NOT ACTIVE/
4021	041214	043111	042511	020104		
4022	041222	052502	020124	047516		
4023	041230	020124	041501	044524		
4024	041236	042526	000			
4025	041241	015	044412	052116	TM13:	.ASCII <CR><LF>/INTERRUPT THE POWER AFTER THE TEST NUMBER APPEARS/
4026	041246	051105	052522	052120		
4027	041254	052040	042510	050040		
4028	041262	053517	051105	040440		
4029	041270	052106	051105	052040		
4030	041276	042510	052040	051505		

4031	041304	020124	052516	041115	
4032	041312	051105	040440	050120	
4033	041320	040505	051522		
4034	041324	044440	020116	044124	.ASCII / IN THE DISPLAY./<CR><LF>
4035	041332	020105	044504	050123	
4036	041340	040514	027131	005015	
4037	041346	043111	054440	052517	.ASCII /IF YOU HAVE AN RD CONSOLE, INTERRUPT THE POWER/
4038	041354	044040	053101	020105	
4039	041362	047101	051040	020104	
4040	041370	047503	051516	046117	
4041	041376	026105	044440	052116	
4042	041404	051105	052522	052120	
4043	041412	052040	042510	050040	
4044	041420	053517	051105		
4045	041424	005015	052101	052040	.ASCII <CR><LF>/AT THE END OF THIS MESSAGE. THEREAFTER, INTERRUPT THE POWER 15(
4046	041432	042510	042440	042116	
4047	041440	047440	020106	044124	
4048	041446	051511	046440	051505	
4049	041454	040523	042507	020056	
4050	041462	052040	042510	042522	
4051	041470	043101	042524	026122	
4052	041476	047111	042524	051122	
4053	041504	050125	020124	044124	
4054	041512	020105	047520	042527	
4055	041520	020122	032461	034050	
4056	041526	020051	047515	042522	
4057	041534	052040	046511	051505	
4058	041542	005015	047524	051040	.ASCIZ <CR><LF>/TO REACH THE END OF SECTION 1./<CR><LF>
4059	041550	040505	044103	052040	
4060	041556	042510	042440	042116	
4061	041564	047440	020106	042523	
4062	041572	052103	047511	020116	
4063	041600	027061	005015	000	
4064	041605	015	005012	047105	TM14: .ASCIZ<CR><LF><LF>/ENTERING SECTION 1/<CR><LF>
4065	041612	042524	044522	043516	
4066	041620	051440	041505	044524	
4067	041626	047117	030440	005015	
4068	041634	000			
4069	041635	015	005012	047105	TM15: .ASCIZ<CR><LF><LF>/ENTERING SECTION 2/<CR><LF>
4070	041642	042524	044522	043516	
4071	041650	051440	041505	044524	
4072	041656	047117	031040	005015	
4073	041664	000			
4074	041665	012	052015	042510	TM76: .ASCIZ<12><15>/THE MASTER IS CPU #/
4075	041672	046440	051501	042524	
4076	041700	020122	051511	041440	
4077	041706	052520	021440	000	
4078	041713	012	006412	042524	TM77: .ASCIZ<12><12><15>/TEST /
4079	041720	052123	000040		
4080	041724	005015	047520	042527	TM100: .ASCIZ <CR><LF>/POWER FAIL CPU # /
4081	041732	020122	040506	046111	
4082	041740	041440	052520	021440	
4083	041746	000040			
4084	041750	005015	047012	020117	TM101: .ASCIZ <CR><LF><LF>/NO MASSBUS DEVICE AVAILABLE ON CPU # /
4085	041756	040515	051523	052502	
4086	041764	020123	042504	044526	

4087 041772 042503 040440 040526
4088 042000 046111 041101 042514
4089 042006 047440 020116 050103
4090 042014 020125 020043 000
4091 042021 015 050012 047522
4092 042026 042503 042105 047111
4093 042034 020107 047524 047040
4094 042042 054105 020124 050103
4095 042050 000125
4096 042052 005015 047520 042527
4097 042060 020122 040506 046111
4098 042066 051125 020105 047117
4099 042074 041440 052520 040440
4100 042102 020123 054105 042520
4101 042110 052103 042105 000
4102 042115 015 043412 052105
4103 042122 051440 052105 052040
4104 042130 020117 047520 042527
4105 042136 020122 040506 046111
4106 042144 042440 052116 051111
4107 042152 020105 054523 052123
4108 042160 046505 027056 056
4109 042165 012 006412 052520
4110 042172 020124 040502 052124
4111 042200 051105 020131 040502
4112 042206 045503 050125 047440
4113 042214 020116 046101 020114
4114 042222 042515 020115 047502
4115 042230 042530 123
4116 042233 012 046415 045501
4117 042240 020105 046101 020114
4118 042246 042515 047515 054522
4119 042254 050040 051117 051524
4120 042262 047440 046116 047111
4121 042270 105
4122 042271 012 046415 045501
4123 042276 020105 046101 020114
4124 042304 050103 020125 047520
4125 042312 042527 026522 050125
4126 042320 051440 044527 041524
4127 042326 042510 020123 051042
4128 042334 047125 047440 020122
4129 042342 040510 052114 042
4130 042347 012 006412 044124
4131 042354 047105 050040 053517
4132 042362 051105 043040 044501
4133 042370 020114 044124 020105
4134 042376 047105 044524 042522
4135 042404 051440 051531 042524
4136 042412 115
4137 042413 012 006412 042522
4138 042420 052123 051117 020105
4139 042426 047520 042527 020122
4140 042434 020065 042523 047503
4141 042442 042116 020123 043101
4142 042450 042524 020122 047520

TM102: .ASCIIZ <CR><LF>/PROCEEDING TO NEXT CPU/

TM103: .ASCIIZ <CR><LF>/POWER FAILURE ON CPU AS EXPECTED/

TM104: .ASCII <CR><LF>/GET SET TO POWER FAIL ENTIRE SYSTEM.../

.ASCII<12><12><15>/PUT BATTERY BACKUP ON ALL MEM BOXES/

.ASCII<12><15>/MAKE ALL MEMORY PORTS ONLINE/

.ASCII<12><15>/MAKE ALL CPU POWER-UP SWITCHES 'RUN OR HALT'/

.ASCII<12><12><15>/THEN POWER FAIL THE ENTIRE SYSTEM/

.ASCII<12><12><15>/RESTORE POWER 5 SECONDS AFTER POWER FAIL/

4143	042456	042527	020122	040506	
4144	042464	046111			
4145	042466	006412	040505	044103	.ASCIZ<12><15>/EACH CPU SHOULD REPORT A POWER FAIL/
4146	042474	041440	052520	051440	
4147	042502	047510	046125	020104	
4148	042510	042522	047520	052122	
4149	042516	040440	050040	053517	
4150	042524	051105	043040	044501	
4151	042532	000114			
4152	042534	005015	043412	052105	TM106: .ASCIZ <CR><LF><LF>/GET SET TO POWER FAIL MEM BOX # /
4153	042542	051440	052105	052040	
4154	042550	020117	047520	042527	
4155	042556	020122	040506	046111	
4156	042564	046440	046505	041040	
4157	042572	054117	021440	000040	
4158	042600	005012	050015	052125	TM107: .ASCII<12><12><15> /PUT BATTERY BACKUP ON ALL MEMORY BOXES/
4159	042606	041040	052101	042524	
4160	042614	054522	041040	041501	
4161	042622	052513	020120	047117	
4162	042630	040440	046114	046440	
4163	042636	046505	051117	020131	
4164	042644	047502	042530	123	
4165	042651	012	046415	045501	.ASCII<12><15>/MAKE ALL MEMORY PORTS ONLINE/
4166	042656	020105	046101	020114	
4167	042664	042515	047515	054522	
4168	042672	050040	051117	051524	
4169	042700	047440	046116	047111	
4170	042706	105			
4171	042707	012	046415	045501	.ASCII<12><15>/MAKE ALL CPU POWER-UP SWITCHES 'RUN OR HALT'/
4172	042714	020105	046101	020114	
4173	042722	050103	020125	047520	
4174	042730	042527	026522	050125	
4175	042736	051440	044527	041524	
4176	042744	042510	020123	051042	
4177	042752	047125	047440	020122	
4178	042760	040510	052114	042	
4179	042765	012	006412	044124	.ASCII<12><12><15>/THEN POWER FAIL THE MEM BOX/
4180	042772	047105	050040	053517	
4181	043000	051105	043040	044501	
4182	043006	020114	044124	020105	
4183	043014	042515	020115	047502	
4184	043022	130			
4185	043023	012	006412	042522	.ASCII<12><12><15>/RESTORE POWER 5 SECONDS AFTER POWER FAIL/
4186	043030	052123	051117	020105	
4187	043036	047520	042527	020122	
4188	043044	020065	042523	047503	
4189	043052	042116	020123	043101	
4190	043060	042524	020122	047520	
4191	043066	042527	020122	040506	
4192	043074	046111			
4193	043076	006412	044124	047105	.ASCII<12><15>/THEN TYPE ANY CHARACTER AT THE MASTER CONSOLE/
4194	043104	052040	050131	020105	
4195	043112	047101	020131	044103	
4196	043120	051101	041501	042524	
4197	043126	020122	052101	052040	
4198	043134	042510	046440	051501	

4199	043142	042524	020122	047503
4200	043150	051516	046117	105
4201	043155	012	042415	041501
4202	043162	020110	050103	020125
4203	043170	044123	052517	042114
4204	043176	051040	050105	051117
4205	043204	020124	020101	047520
4206	043212	042527	020122	040506
4207	043220	046111	051125	000105
4208	043226	005012	050015	052125
4209	043234	041040	052101	042524
4210	043242	054522	041040	041501
4211	043250	052513	020120	047117
4212	043256	047440	046114	046440
4213	043264	046505	051117	020131
4214	043272	047502	042530	123
4215	043277	012	046415	045501
4216	043304	020105	046101	020114
4217	043312	042515	047515	054522
4218	043320	050040	051117	051524
4219	043326	047440	043106	044514
4220	043334	042516	047440	020116
4221	043342	042515	020115	047502
4222	043350	020130	047524	041040
4223	043356	020105	047520	042527
4224	043364	026522	040506	046111
4225	043372	042105		
4226	043374	006412	040515	042513
4227	043402	040440	046114	041440
4228	043410	052520	050040	053517
4229	043416	051105	052455	020120
4230	043424	053523	052111	044103
4231	043432	051505	021040	052522
4232	043440	020116	051117	044040
4233	043446	046101	021124	
4234	043452	005012	052015	042510
4235	043460	020116	047520	042527
4236	043466	020122	040506	046111
4237	043474	052040	042510	046440
4238	043502	046505	041040	054117
4239	043510	005012	051015	051505
4240	043516	047524	042522	050040
4241	043524	053517	051105	032440
4242	043532	051440	041505	047117
4243	043540	051504	040440	052106
4244	043546	051105	050040	053517
4245	043554	051105	043040	044501
4246	043562	114		
4247	043563	012	051015	051505
4248	043570	047524	042522	040440
4249	043576	046114	046440	046505
4250	043604	051117	020131	047520
4251	043612	052122	020123	047117
4252	043620	044514	042516	
4253	043624	006412	044124	047105
4254	043632	052040	050131	020105

.ASCIZ<12><15>/EACH CPU SHOULD REPORT A POWER FAILURE/

TM108: .ASCII<12><12><15>/PUT BATTERY BACKUP ON ALL MEMORY BOXFS/

.ASCII<12><15>/MAKE ALL MEMORY PORTS OFFLINE ON MEM BOX TO BE POWER-FAILED/

.ASCII<12><15>/MAKE ALL CPU POWER-UP SWITCHES 'RUN OR HALT'/

.ASCII<12><12><15>/THEN POWER FAIL THE MEM BOX/

.ASCII<12><12><15>/RESTORE POWER 5 SECONDS AFTER POWER FAIL/

.ASCII<12><15>/RESTORE ALL MEMORY PORTS ONLINE/

.ASCII<12><15>/THEN TYPE ANY CHARACTER AT THE MASTER CONSOLE/

4255	043640	047101	020131	044103	
4256	043646	051101	041501	042524	
4257	043654	020122	052101	052040	
4258	043662	042510	046440	051501	
4259	043670	042524	020122	047503	
4260	043676	051516	046117	105	
4261	043703	012	047015	020117	.ASCIZ<12><15>/NO CPU SHOULD REPORT A POWER FAILURE/
4262	043710	050103	020125	044123	
4263	043716	052517	042114	051040	
4264	043724	050105	051117	020124	
4265	043732	020101	047520	042527	
4266	043740	020122	040506	046111	
4267	043746	051125	000105		
4268	043752	006412	053412	051101	\$PGM1: .ASCIZ <12><15><12>/WARNING: DRIVE # /
4269	043760	044516	043516	020072	
4270	043766	042040	044522	042526	
4271	043774	021440	000040		
4272	044000	006412	047117	041440	\$PGM2: .ASCIZ<12><15> /ON CPU # /
4273	044006	052520	021440	000040	
4274	044014	006412	051511	050040	\$PGM3: .ASCII <12><15>/IS PROGRAMMABLE OVER BOTH A AND B PORTS/
4275	044022	047522	051107	046501	
4276	044030	040515	046102	020105	
4277	044036	053117	051105	041040	
4278	044044	052117	020110	020101	
4279	044052	047101	020104	020102	
4280	044060	047520	052122	123	
4281	044065	012	052015	042510	.ASCIZ<12><15>/THE DRIVE WILL BE USED LATER IN THIS DIAGNOSTIC/
4282	044072	042040	044522	042526	
4283	044100	053440	046111	020114	
4284	044106	042502	052440	042523	
4285	044114	020104	040514	042524	
4286	044122	020122	047111	052040	
4287	044130	044510	020123	044504	
4288	044136	043501	047516	052123	
4289	044144	041511	000		
4290	044147	012	047015	020117	NODEV: .ASCIZ <12><15>/NO MASSBUS DEVICES ON CPU # /
4291	044154	040515	051523	052502	
4292	044162	020123	042504	044526	
4293	044170	042503	020123	047117	
4294	044176	041440	052520	021440	
4295	044204	000040			
4296	044206	005012	047415	046116	TM109: .ASCIZ <12><12><15>/ONLY ONE MEMORY BOX ONLINE- SKIPPING THIS TEST/
4297	044214	020131	047117	020105	
4298	044222	042515	047515	054522	
4299	044230	041040	054117	047440	
4300	044236	046116	047111	026505	
4301	044244	051440	044513	050120	
4302	044252	047111	020107	044124	
4303	044260	051511	052040	051505	
4304	044266	000124			
4305	044270	005012	042015	051511	TM110: .ASCII<12><12><15>/DISABLE BATTERY BACKUP ON MEM BOX TO BE POWER-FAILED/
4306	044276	041101	042514	041040	
4307	044304	052101	042524	054522	
4308	044312	041040	041501	052513	
4309	044320	020120	047117	046440	
4310	044326	046505	041040	051117	

4311	044334	052040	020117	042502
4312	044342	050040	053517	051105
4313	044350	043055	044501	042514
4314	044356	104		
4315	044357	012	050015	052125
4316	044364	040440	046114	051440
4317	044372	040514	042526	041440
4318	044400	052520	046440	046505
4319	044406	050040	051117	051524
4320	044414	047440	046116	047111
4321	044422	105		
4322	044423	012	046415	045501
4323	044430	020105	040515	052123
4324	044436	051105	041440	052520
4325	044444	046440	046505	050040
4326	044452	051117	020124	043117
4327	044460	046106	047111	020105
4328	044466	047117	041040	054117
4329	044474	052040	020117	042502
4330	044502	050040	053517	051105
4331	044510	043055	044501	042514
4332	044516	104		
4333	044517	012	046415	045501
4334	044524	020105	046101	020114
4335	044532	050103	020125	047520
4336	044540	042527	026522	050125
4337	044546	051440	044527	041524
4338	044554	042510	020123	051042
4339	044562	047125	047440	020122
4340	044570	047502	052117	042
4341	044575	012	006412	044124
4342	044602	047105	050040	053517
4343	044610	051105	043040	044501
4344	044616	020114	044124	020105
4345	044624	042515	020115	047502
4346	044632	130		
4347	044633	012	006412	042522
4348	044640	052123	051117	020105
4349	044646	047520	042527	020122
4350	044654	020065	042523	047503
4351	044662	042116	020123	043101
4352	044670	042524	020122	047520
4353	044676	042527	020122	040506
4354	044704	046111		
4355	044706	006412	042522	052123
4356	044714	051117	020105	046101
4357	044722	020114	042515	020115
4358	044730	047520	052122	020123
4359	044736	047117	044514	042516
4360	044744	006412	042522	052123
4361	044752	051117	020105	046101
4362	044760	020114	050103	020125
4363	044766	047520	042527	026522
4364	044774	050125	051440	044527
4365	045002	041524	042510	020123
4366	045010	047524	021040	052522

.ASCII<12><15>/PUT ALL SLAVE CPU MEM PORTS ONLINE/

.ASCII<12><15>/MAKE MASTER CPU MEM PORT OFFLINE ON BOX TO BE POWER-FAILED/

.ASCII<12><15>/MAKE ALL CPU POWER-UP SWITCHES 'RUN OR BOOT'/

.ASCII<12><12><15>/THEN POWER FAIL THE MEM BOX/

.ASCII<12><12><15>/RESTORE POWER 5 SECONDS AFTER POWER FAIL/

.ASCII<12><15>/RESTORE ALL MEM PORTS ONLINE/

.ASCII<12><15>/RESTORE ALL CPU POWER-UP SWITCHES TO 'RUN OR HALT'/

4367	045016	020116	051117	044040	
4368	045024	046101	021124		
4369	045030	006412	044124	047105	.ASCII<12><15>/THEN TYPE ANY CHARACTER AT THE MASTER CONSOLE/
4370	045036	052040	050131	020105	
4371	045044	047101	020131	044103	
4372	045052	051101	041501	042524	
4373	045060	020122	052101	052040	
4374	045066	042510	046440	051501	
4375	045074	042524	020122	047503	
4376	045102	051516	046117	105	
4377	045107	012	042415	041501	.ASCII<12><15>/EACH SLAVE CPU SHOULD REPORT AN INTERRUPT/
4378	045114	020110	046123	053101	
4379	045122	020105	050103	020125	
4380	045130	044123	052517	042114	
4381	045136	051040	050105	051117	
4382	045144	020124	047101	044440	
4383	045152	052116	051105	052522	
4384	045160	052120	000		
4385	045163	012	041415	052520	TM111: .ASCII<12><15>/CPU INTERRUPTED AS EXPECTED/
4386	045170	044440	052116	051105	
4387	045176	052522	052120	042105	
4388	045204	040440	020123	054105	
4389	045212	042520	052103	042105	
4390	045220	000			
4391	045221	012	050015	053517	\$DOWN: .ASCII <12><15>/POWER DOWN TIME WAS UNDER 2 MILISECONDS /
4392	045226	051105	042040	053517	
4393	045234	020116	044524	042515	
4394	045242	053440	051501	052440	
4395	045250	042116	051105	031040	
4396	045256	046440	046111	051511	
4397	045264	041505	047117	051504	
4398	045272	000040			
4399	045274	005015	047125	054105	EM1: .ASCII <CR><LF>/UNEXPECTED POWER FAILURE ON CPU/
4400	045302	042520	052103	042105	
4401	045310	050040	053517	051105	
4402	045316	043040	044501	052514	
4403	045324	042522	047440	020116	
4404	045332	050103	000125		
4405	045336	005015	047125	054105	EM2: .ASCII <CR><LF>/UNEXPECTED POWER UP SEQUENCE ON CPU/
4406	045344	042520	052103	042105	
4407	045352	050040	053517	051105	
4408	045360	052440	020120	042523	
4409	045366	052521	047105	042503	
4410	045374	047440	020116	050103	
4411	045402	000125			
4412	045404	005015	046111	042514	EM3: .ASCII <CR><LF>/ILLEGAL POWER DOWN SEQUENCE ON CPU/
4413	045412	040507	020114	047520	
4414	045420	042527	020122	047504	
4415	045426	047127	051440	050505	
4416	045434	042525	041516	020105	
4417	045442	047117	041440	052520	
4418	045450	000			
4419	045451	015	044412	046114	EM4: .ASCII <CR><LF>/ILLEGAL POWER UP SEQUENCE/
4420	045456	043505	046101	050040	
4421	045464	053517	051105	052440	
4422	045472	020120	042523	052521	

4423	045500	047105	042503	000		
4424	045505	015	052412	042516	EM5:	.ASCIZ <CR><LF>/UNEXPECTED TRAP TO 4/
4425	045512	050130	041505	042524		
4426	045520	020104	051124	050101		
4427	045526	052040	020117	000064		
4428	045534	005015	047125	054105	EM6:	.ASCIZ <CR><LF>/UNEXPECTED TRAP TO 10/
4429	045542	042520	052103	042105		
4430	045550	052040	040522	020120		
4431	045556	047524	030440	000060		
4432	045564	005015	047125	054105	EM7:	.ASCIZ <CR><LF>/UNEXPECTED TRAP TO 114/
4433	045572	042520	052103	042105		
4434	045600	052040	040522	020120		
4435	045606	047524	030440	032061		
4436	045614	000				
4437	045615	015	040412	042104	EM10:	.ASCIZ <CR><LF>/ADDRESS ON STACK IS WRONG/
4438	045622	042522	051523	047440		
4439	045630	020116	052123	041501		
4440	045636	020113	051511	053440		
4441	045644	047522	043516	000		
4442	045651	015	047412	042114	EM11:	.ASCIZ <CR><LF>/OLD PS IS WRONG/
4443	045656	050040	020123	051511		
4444	045664	053440	047522	043516		
4445	045672	000				
4446	045673	015	047412	042104	EM12:	.ASCIZ <CR><LF>/ODD ADDRESS TRAP FAILED/
4447	045700	040440	042104	042522		
4448	045706	051523	052040	040522		
4449	045714	020120	040506	046111		
4450	045722	042105	000			
4451	045725	015	046412	046505	EM13:	.ASCIZ <CR><LF>/MEMORY CORRUPTED ON POWER FAIL/
4452	045732	051117	020131	047503		
4453	045740	051122	050125	042524		
4454	045746	020104	047117	050040		
4455	045754	053517	051105	043040		
4456	045762	044501	000114			
4457	045766	005015	044524	042515	EM14:	.ASCIZ <CR><LF>/TIMEOUT TRAP FAILED/
4458	045774	052517	020124	051124		
4459	046002	050101	043040	044501		
4460	046010	042514	000104			
4461	046014	005015	047520	042527	EM15:	.ASCIZ <CR><LF>/POWER FAIL RETURNED TOO SOON/
4462	046022	020122	040506	046111		
4463	046030	051040	052105	051125		
4464	046036	042516	020104	047524		
4465	046044	020117	047523	047117		
4466	046052	000				
4467	046053	015	047012	052117	EM16:	.ASCIZ <CR><LF>/NOT ENOUGH OR TOO MANY INSTRUCTIONS EXECUTED/
4468	046060	042440	047516	043525		
4469	046066	020110	051117	052040		
4470	046074	047517	046440	047101		
4471	046102	020131	047111	052123		
4472	046110	052522	052103	047511		
4473	046116	051516	042440	042530		
4474	046124	052503	042524	000104		
4475	046132	005015	047516	046440	EM17:	.ASCIZ <CR><LF>/NO MEM. MANG. VIOLATION OR TRAP TO 4/
4476	046140	046505	020056	040515		
4477	046146	043516	020056	044526		
4478	046154	046117	052101	047511		

4479	046162	020116	051117	052040					
4480	046170	040522	020120	047524					
4481	046176	032040	000						
4482	046201	015	047012	020117	EM20:	.ASCIZ	<CR><LF>/NO IIST INTERRUPT/		
4483	046206	044511	052123	044440					
4484	046214	052116	051105	052522					
4485	046222	052120	000						
4486	046225	015	044412	041516	EM21:	.ASCIZ	<CR><LF>?INCORRECT BRK AND/OR DCF FLAGS?		
4487	046232	051117	042522	052103					
4488	046240	041040	045522	040440					
4489	046246	042116	047457	020122					
4490	046254	041504	020106	046106					
4491	046262	043501	000123						
4492	046266	005015	050103	020125	EM22:	.ASCIZ	<CR><LF>/CPU DID NOT TRAP TO VIRTUAL 24/		
4493	046274	044504	020104	047516					
4494	046302	020124	051124	050101					
4495	046310	052040	020117	044526					
4496	046316	052122	040525	020114					
4497	046324	032062	000						
4498	046327	015	041412	042510	EM23:	.ASCIZ	<CR><LF>/CHECKSUM ON MASSBUS TRANSFER IS WRONG/		
4499	046334	045503	052523	020115					
4500	046342	047117	046440	051501					
4501	046350	041123	051525	052040					
4502	046356	040522	051516	042506					
4503	046364	020122	051511	053440					
4504	046372	047522	043516	000					
4505	046377	012	047015	020117	EM24:	.ASCIZ	<12><15>/NO POWER FAIL ON CPU/		
4506	046404	047520	042527	020122					
4507	046412	040506	046111	047440					
4508	046420	020116	050103	000125					
4509	046426	006412	047125	054105	EM25:	.ASCIZ	<12><15>/UNEXPECTED CPU INTERRUPT/		
4510	046434	042520	052103	042105					
4511	046442	041440	052520	044440					
4512	046450	052116	051105	052522					
4513	046456	052120	000						
4514									
4515	046461	015	044412	051511	DH5:	.ASCIZ	<CR><LF>/IISTID PC CPUERR/		
4516	046466	044524	004504	020040					
4517	046474	041520	020040	020040					
4518	046502	020040	041440	052520					
4519	046510	051105	000122						
4520	046514	005015	020040	044511	DH7:	.ASCIZ	<CR><LF>/ IISTID ERRORPC CPUERR MEMERR/		
4521	046522	052123	042111	020040					
4522	046530	020040	051105	047522					
4523	046536	050122	020103	041440					
4524	046544	052520	051105	020122					
4525	046552	020040	042515	042515					
4526	046560	051122	000						
4527	046563	015	052012	051505	DH10:	.ASCIZ	<CR><LF>/TESTNO ERRORPC/		
4528	046570	047124	020117	020040					
4529	046576	051105	047522	050122					
4530	046604	000103							
4531	046606	005015	042524	052123	DH11:	.ASCIZ	<CR><LF>/TESTNO ERRORPC PS/		
4532	046614	047516	020040	042440					
4533	046622	051122	051117	041520					
4534	046630	020040	020040	050040					

4535	046636	000123							
4536	046640	006412	042524	052123	DH12:	.ASCIZ	<12><15>/TESTNO	ERRORPC	PAGE ADDRESS REGISTER OF BAD MEMORY/
4537	046646	047516	042411	051122					
4538	046654	051117	041520	020040					
4539	046662	050040	043501	020105					
4540	046670	042101	051104	051505					
4541	046676	020123	042522	044507					
4542	046704	052123	051105	047440					
4543	046712	020106	040502	020104					
4544	046720	042515	047515	054522					
4545	046726	000							
4546	046727	015	052012	051505	DH14:	.ASCIZ	<CR><LF>/TESTNO	ERRORPC	CPUERR/
4547	046734	047124	020117	020040					
4548	046742	051105	047522	050122					
4549	046750	020103	041440	052520					
4550	046756	051105	000122						
4551	046762	005015	042524	052123	DH20:	.ASCIZ	<CR><LF>/TESTNO	IISTID	ACR PGTE PGCS/
4552	046770	047516	020040	020040					
4553	046776	044511	052123	042111					
4554	047004	020040	020040	040440					
4555	047012	051103	020040	020040					
4556	047020	050040	052107	020105					
4557	047026	020040	020040	043520					
4558	047034	051503	000						
4559	047037	015	052012	051505	DH21:	.ASCIZ	<CR><LF>/TESTNO	IISTID	FOUND SHOULD BE/
4560	047044	047124	020117	020040					
4561	047052	044440	051511	044524					
4562	047060	020104	020040	047506					
4563	047066	047125	020104	020040					
4564	047074	051440	047510	046125					
4565	047102	020104	042502	000					
4566	047107	015	052012	051505	DH22:	.ASCIZ	<CR><LF>/TESTNO	IISTID	ERRORPC/
4567	047114	047124	020117	020040					
4568	047122	044511	052123	042111					
4569	047130	020040	020040	051105					
4570	047136	047522	050122	000103					
4571							.EVEN		
4572									
4573	047144	014240	014064	014250	DT5:	\$REG0,\$ERRPC,\$REG1,0			
4574	047152	000000							
4575	047154	014240	014064	014250	DT7:	\$REG0,\$ERRPC,\$REG1,\$REG2,0			
4576	047162	014260	000000						
4577	047166	014002	014064	000000	DT10:	\$STSTM,\$ERRPC,0			
4578	047174	014002	014064	014240	DT11:	\$STSTM,\$ERRPC,\$REG0,0			
4579	047202	000000							
4580	047204	014002	014064	014250	DT12:	\$STSTM,\$ERRPC,\$REG1,0			
4581	047212	000000							
4582	047214	014002	014240	014250	DT20:	\$STSTM,\$REG0,\$REG1,\$REG2,\$REG3,0			
4583	047222	014260	014270	000000					
4584	047230	014002	014240	014250	DT21:	\$STSTM,\$REG0,\$REG1,\$REG2,0			
4585	047236	014260	000000						
4586	047242	014002	014240	014064	DT22:	\$STSTM,\$REG0,\$ERRPC,0			
4587	047250	000000							
4588	047252	005015	020133	045517	OK:	.ASCIZ	<CR><LF>/[OK] /		
4589	047260	056440	020040	000					
4590		047266					.EVEN		

4591	047266	001000	ERRBUF: .BLKW	1000	:ERROR ASCII MSG STORAGE AREA
4592	051266	020000	DSKBUF: .BLKW	20000	:MASSBUS BUFFER AREA
4593	111266	000000	END: C		
4594					
4595		000001	.END		

MAPH15=	170266	421#				
MAPH16=	170272	423#				
MAPH17=	170276	425#				
MAPH2 =	170212	463#				
MAPH20=	170302	427#				
MAPH21=	170306	429#				
MAPH22=	170312	431#				
MAPH23=	170316	433#				
MAPH24=	170320	435#				
MAPH25=	170326	437#				
MAPH26=	170332	439#				
MAPH27=	170336	441#				
MAPH3 =	170216	465#				
MAPH30=	170342	443#				
MAPH31=	170346	445#				
MAPH32=	170352	447#				
MAPH33=	170356	449#				
MAPH34=	170362	451#				
MAPH35=	170366	453#				
MAPH36=	170372	455#				
MAPH37=	170376	457#	3047	3090	3191	3234
MAPH4 =	170222	467#				
MAPH5 =	170226	469#				
MAPH6 =	170232	471#				
MAPH7 =	170236	473#				
MAPL0 =	170200	458#				
MAPL00=	170200	394#	458	3045	3092	3189 3236
MAPL01=	170204	396#	460			
MAPL02=	170210	398#	462			
MAPL03=	170214	400#	464			
MAPL04=	170220	402#	466			
MAPL05=	170224	404#	468			
MAPL06=	170230	406#	470			
MAPL07=	170234	408#	472			
MAPL1 =	170204	460#				
MAPL10=	170240	410#				
MAPL11=	170244	412#				
MAPL12=	170250	414#				
MAPL13=	170254	416#				
MAPL14=	170260	418#				
MAPL15=	170264	420#				
MAPL16=	170270	422#				
MAPL17=	170274	424#				
MAPL2 =	170210	462#				
MAPL20=	170300	426#				
MAPL21=	170304	428#				
MAPL22=	170310	430#				
MAPL23=	170314	432#				
MAPL24=	170320	434#				
MAPL25=	170324	436#				
MAPL26=	170330	438#				
MAPL27=	170334	440#				
MAPL3 =	170214	464#				
MAPL30=	170340	442#				
MAPL31=	170344	444#				
MAPL32=	170350	446#				

TM11	041176	1158	4018#																
TM110	044270	2707	4305#																
TM111	045163	2024	2933	4385#															
TM12	041207	1161	4020#																
TM13	041241	1317	4025#																
TM14	041605	1314	4064#																
TM15	041635	1826	4069#																
TM2	040676	1108	3983#																
TM4	040723	1117	3987#																
TM5	040765	1120	3993#																
TM6	041033	1102	4000#																
TM7	041100	1100	4007#																
TM76	041665	1136	4074#																
TM77	041713	1955	2078	2163	2334	2496	2657	2779	4078#										
TPVEC =	000064	211#																	
TQL1	017024	880#	3841*	3857*	3867*	3907*													
TRAPVE=	000034	209#	1065*	1066*															
TRTVEC=	000014	204#																	
TST1	021450	1229	1230	1231	1232	1240	1244	1248	1302#	1350	3579	3580	3581	3582					
TST10	022762	1560#	1600																
TST11	023160	1606#	1624																
TST12	023252	1630#	1654																
TST13	023406	1660#	1684																
TST14	023542	1690#	1714																
TST15	023676	1720#	1744																
TST16	024032	1750#	1778																
TST17	024674	1899	1916#	1971	1976	2008	2033												
TST2	021704	1356#	1388																
TST20	025524	1934	2042#	2095	2116														
TST21	026114	2059	2124#	2185	2202	2229	2237	2268											
TST22	027076	2141	2280#																
TST23	030120	2305	2345	2357	2442#														
TST24	031142	2467	2507	2519	2603#														
TST25	032164	2628	2668	2680	2765#														
TST3	022044	1392#	1421																
TST4	022176	1427#	1455																
TST5	022330	1462#	1483																
TST6	022444	1491#	1525																
TST7	022636	1531#	1554																
TS17A	025050	1943	1951#																
TS17B	025172	1945	1979#	1981															
TS20A	025674	2067	2074#																
TS20B	026006	2069	2097#	2099															
TS21A	026270	2150	2157#																
TS21B	026656	2152	2230#																
TS22A	027352	2294	2331#																
TS22B	027162	2293	2296#	2314	2316	2319	2324	2327											
TS23A	030374	2456	2493#																
TS23B	030204	2455	2458#	2476	2478	2481	2486	2489											
TS24A	031416	2617	2654#																
TS24B	031226	2616	2619#	2637	2639	2642	2647	2650	2934										
TS25A	032232	2774	2776#																
TS25B	032264	2775	2787#																
TYPDS =	104405	1138	1160	1874	1889	1892	1963	2086	2181	2192	2382	2544	2705	3320					
TYPE =	104401	3335	3344	3353	3362	3370	3961#												
		1#	1076	1100	1102	1108	1111	1112	1117	1120	1136	1157	1158	1161					

.S40CA 1#
.1170 1# 90

. ABS. 111270 000

ERRORS DETECTED: 0

CEKBGC.BIN,CEKBGC.LST/CRF/SOL/NL:TOC=CEKBGC.SML,CEKBGC.P11
RUN-TIME: 62 58 5 SECONDS
RUN-TIME RATIO: 196/127=1.5
CORE USED: 39K (77 PAGES)